UNIVERSIDADE FEDERAL DE JATAÍ (UFJ)
INSTITUTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MICHEL OLIVEIRA DA SILVA

Mineração de Processos em WorkFlow Nets Interorganizacionais

JATAÍ-GOIÁS
2025

MICHEL OLIVEIRA DA SILVA

Mineração de Processos em WorkFlow Nets Interorganizacionais

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação, Instituto de Ciências Exatas e Tecnológicas, da Universidade Federal de Jataí (UFJ), como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientadora: Profª. Dra. Franciny Medeiros Barreto

JATAÍ-GOIÁS
2025

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UFJ.

Silva, Michel Oliveira da

Mineração de Processos em WorkFlow Nets Interorganizacionais / Michel Oliveira da Silva. - 2025.

77 f.: il.

Orientadora: Profª. Dr ª. Franciny Medeiros Barreto

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Jataí, Instituto de Ciências Exatas e Tecnológicas, Bacharelado em Ciência da Computação, Jataí, 2025.

Inclui siglas, símbolos, tabelas, algoritmos, lista de figuras, lista de tabelas.

1. Mineração de Processos. 2. Redes de Petri Coloridas. 3. WorkFlow nets Interorganizacionais. I. Barreto, Franciny Medeiros, orient. II. Título.

CDU 004

UNIVERSIDADE FEDERAL DE JATAÍ
INSTITUTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS

DECLARAÇÃO DE APROVAÇÃO DA VERSÃO FINAL

Declaro que o discente Michel Oliveira da Silva do curso de Ciência da Computação foi aprovado na defesa do Trabalho de Conclusão de Curso (TCC) com o título final Mineração de Processos em WorkFlow Nets Interorganizacionais na data de 05/12/2025 e efetuou todas as correções pertinentes sugeridas pela banca examinadora, composta pelo seguintes membros:

Orientador(a)	Franciny Medeiros Barreto
Membro 1	Críscilla Maia Costa Rezende
Membro 2	Joslaine Cristina Jeske de Freitas

Declaro ainda que a versão final anexada a este processo está adequada para ser devidamente depositada em repositório institucional.

Franciny Medeiros Barreto

Professora do curso de Ciência da Computação

Observação

Esta declaração deve ser assinada pelo(a) orientador(a)

Documento assinado eletronicamente por **FRANCINY MEDEIROS BARRETO**, **Professor(a) do Magistério Superior**, em 08/12/2025, às 16:07, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufj.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0519394** e o código CRC

Referência: Processo nº 23854.009875/2025-98

SEI nº 0519394

Resumo

O atual cenário de globalização impulsionou a fragmentação das cadeias produtivas, elevando a adoção de processos interorganizacionais. A complexidade presente nesses fluxos, que envolvem múltiplas organizações independentes, torna desafiadora a modelagem e a análise. Embora as *WorkFlow nets* interorganizacionais (IOWF-*nets*) baseadas em redes de Petri ofereçam um formalismo robusto para essa representação, a literatura existente frequentemente exige um domínio matemático avançado para a verificação de propriedades. Este trabalho propõe um mecanismo de análise que combina a modelagem em redes de Petri Coloridas e mineração de processos. A metodologia consistiu na modelagem de um processo de compras interorganizacional utilizando a ferramenta CPN Tools, a qual foi adaptada através da inserção de funções em linguagem CPN ML para a geração automática de *event logs* no formato MXML. Os dados gerados pela simulação foram posteriormente unificados e analisados na ferramenta ProM. Os resultados demonstraram que, embora o modelo fosse estruturalmente correto sob a ótica das redes de Petri, a mineração de processos revelou inconsistências semânticas e de desempenho, como atividades executadas fora de ordem e gargalos temporais invisíveis na modelagem estática. Conclui-se que a abordagem com o mecanismo proposto facilita a identificação de falhas em processos distribuídos, permitindo uma validação mais assertiva antes da implementação real ou tomada de decisão.

Palavras-chave: *Mineração de Processos; Redes de Petri Coloridas; WorkFlow nets Interorganizacionais.*

Abstract

The current globalization scenario has driven the fragmentation of production chains, increasing the adoption of inter-organizational processes. The complexity inherent in these flows, which involve multiple independent organizations, makes modeling and analysis challenging. Although inter-organizational WorkFlow nets (IOWF-nets) based on Petri nets offer a robust formalism for this representation, existing literature often requires advanced mathematical expertise for property verification. This work proposes an analysis mechanism that combines modeling in Coloured Petri Nets and process mining. The methodology consisted of modeling an inter-organizational purchasing process using CPN Tools, which was adapted through the insertion of CPN ML functions for the automatic generation of event logs in MXML format. The data generated by the simulation were subsequently unified and analyzed using the ProM tool. The results demonstrated that, although the model was structurally correct from a Petri net perspective, process mining revealed semantic and performance inconsistencies, such as out-of-order activities and temporal bottlenecks invisible in static modeling. It is concluded that the approach using the proposed mechanism facilitates the identification of failures in distributed processes, allowing for more accurate validation prior to real-world implementation or decision-making.

Keywords: *Process Mining; Coloured Petri Nets; Inter-organizational WorkFlow nets.*

Dedico este trabalho para as pessoas mais importantes da minha vida, àquelas que me deram a oportunidade de contemplar a dádiva da vida, meus pais. À minha mãe que sempre sonhou e acreditou junto comigo que fosse possível, mesmo com todas as barreiras enfrentadas pelo caminho, demonstrando não apenas em palavras mas em atitudes, lutando dia após dia para garantir a mim esta oportunidade. Ao meu pai que sempre fez o possível e o impossível para que nunca faltasse nada para nossa família, nos honrando com sua bravura e coragem para garantir a segurança e o bem-estar familiar. Dedico também a minha irmã mais nova, que por muitas vezes apesar da pouca idade, demonstrou sua maturidade através do seu apoio em momentos obscuros dessa caminhada. Prolongo esta dedicatória a todas as pessoas que se encontram em dúvida sobre sua própria capacidade, mas que mesmo diante de tudo persistem e escrevem cada capítulo de sua jornada.

Agradecimentos

Primeiramente agradeço a Deus por todas as conquistas até aqui, e pela oportunidade de viver este momento. Agradeço à minha família que sempre me apoiou e sonhou junto comigo, agradeço à minha orientadora que me acompanhou durante todos os momentos, fornecendo todo o apoio necessário para que fosse possível a realização deste trabalho, que foi o ponto chave para que esta pesquisa fosse realizada com excelência. Agradeço pelos momentos vividos e pela amizade de toda a turma 14 do curso de Ciência da Computação, em especial para meus grandes amigos que fazem parte do nosso grupo intitulado "Os de Verdade". Destaco o apreço em especial para meus grandes amigos Gabriel Oka Duarte e Pedro Henrique Gomides Moraes, que sempre me motivaram e incentivaram a buscar pelo crescimento em ambiente acadêmico e pessoal.

"O agregado de nossa alegria e sofrimento, milhares de religiões, ideologias e doutrinas econômicas, cada caçador e coletor, cada herói e covarde, cada criador e destruidor da civilização, cada rei e camponês, cada jovem casal apaixonado, cada mãe e pai, cada criança esperançosa, inventor e explorador, cada mestre de moral, cada político corrupto, cada superastro, cada líder supremo, cada santo e pecador na história da nossa espécie viveu aqui — num grão de poeira suspenso num raio de sol.

CARL SAGAN — Pálido Ponto Azul

Lista de ilustrações

Figura 1 – Elementos básicos da rede de Petri. 18
Figura 2 – Modelo de funcionamento de um semáforo em rede de Petri. 19
Figura 3 – Funcionamento da rede que representa um semáforo. 20
Figura 4 – Exemplo de um processo de negócio modelado por uma *WorkFlow net*. 21
Figura 5 – Roteamento sequencial em uma WF-*net*. 23
Figura 6 – Exemplo do roteamento paralelo em uma WF-*net*. 23
Figura 7 – Exemplo de um roteamento condicional em uma WF-*net*. 24
Figura 8 – Exemplo de um roteamento iterativo em uma WF-*net*. 24
Figura 9 – Exemplo de uma WF-*net* adaptada para a verificação da propriedades *Soundness*. 26
Figura 10 – Exemplo de uma IOWF-*net Sound* local. 28
Figura 11 – Exemplo de uma IOWF-*net Sound* global. 29
Figura 12 – Exemplo de uma IOWF-*net Sound*. 30
Figura 13 – Exemplo de uma rede de Petri Colorida. 33
Figura 14 – *Screenshot* das ferramentas do CPN Tools. 34
Figura 15 – *Screenshot* de um modelo criado no CPN Tools. 35
Figura 16 – IOWF-*net* adaptada de Aalst (1999). 46
Figura 17 – IOWF-*net* com os tipos de dados associados representando as informações do processo. 48
Figura 18 – Seção de declarações da ferramenta CPN Tools. 55
Figura 19 – Estado inicial da rede criada no CPN Tools. 57
Figura 20 – IOWF-*net* com funções geradoras de *logs* inseridas. 58
Figura 21 – *Screenshot* da interface do *ProM Imports Framework*. 62
Figura 22 – *Screenshot* da aba de propriedades do filtro. 62
Figura 23 – *Screenshot* da interface da ferramenta ProM. 63
Figura 24 – *Screenshot* da janela de seleção do *plugin* de importação. 63
Figura 25 – *Screenshot* da ferramenta ProM após inserir o arquivo no formato MXML. 64
Figura 26 – *Screenshot* da ferramenta ProM durante a escolha do *plugin*. 64
Figura 27 – *Screenshot* da ferramenta ProM ao executar o *plugin* para o modelo da *WorkFlow net* Interorganizacional. 65
Figura 28 – Principais elementos do *launcher* do *plugin*. 67
Figura 29 – *Screenshot* do filtro caminhos e desvios. 68
Figura 30 – *Screenshot* do filtro caminhos e tempos de permanência. 69
Figura 31 – *Screenshot* do filtro alternador da visualização de traço. 69
Figura 32 – Análise das atividades de aprovar e reavaliar requisição. 71
Figura 33 – *Screenshot* do filtro alternador da visualização de traço de um caso isolado. 72

Lista de tabelas

Tabela 1 – Exemplo de *log* com variações no processo (anomalias e desvios). . . . 37
Tabela 2 – Resumo dos conceitos de *Play-in*, *Play-out* e *Replay*. 40
Tabela 3 – Comparação dos trabalhos relacionados. 44
Tabela 4 – Resumo da troca de nomes das transições do modelo da Figura 12 para a Figura 16 47

Sumário

1 Introdução 14
INTRODUÇÃO 14
2 Referencial Teórico 17
2.1 REDES DE PETRI 17
2.1.1 WorkFlow nets 21
2.1.2 WorkFlow net interorganizacional 26
2.1.3 Redes de Petri Coloridas 31
2.1.4 CPN Tools 33
2.2 MINERAÇÃO DE PROCESSOS 36
3 Trabalhos relacionados 41
3.1 LEVANTAMENTO BIBLIOGRÁFICO 41
3.2 CONSIDERAÇÕES FINAIS 43
4 Modelo WorkFlow net Interorganizacional 45
4.1 ESCOLHA DO MODELO 45
4.2 MODELO IOWF-*NET* DE SIMULAÇÃO 45
4.2.1 Funções CPN ML do Modelo de Simulação 49
4.2.2 Declarações dos *color sets* 50
4.2.3 Declarações das fichas 52
4.2.4 Declarações de Variáveis 53
4.2.5 Declarações de *Logs* 54
4.3 INSERÇÃO DAS FUNÇÕES GERADORAS DE *LOGS* 57
4.3.1 Configuração de *Log* e Transições 58
4.3.2 Funções de Registro (*Logging Functions*) 58
5 Análise do Modelo 61
5.1 MINERAÇÃO DE PROCESSOS 61
5.1.1 ProM Imports Framework 61
5.1.2 ProM 63
5.1.3 Análise e Resultados 70
6 Conclusões e Trabalhos Futuros 73
6.1 CONCLUSÕES 73
6.2 TRABALHOS FUTUROS 73

Referências 75

1 Introdução

Devido ao crescimento da globalização, o processo de industrialização sofreu algumas interferências que foram responsáveis por mudar a forma como diversos produtos são fabricados, pois 70% do comércio global ocorre por meio de cadeias de valor globais (OECD, 2024). Isso pode ser evidenciado através da globalização produtiva que foi intensificada a partir da década de 90, ocasionando o deslocamento industrial para países subdesenvolvidos onde se tem mão de obra de menor valor, o que deu início a fragmentação das cadeias produtivas (Relações Exteriores, 2025). A partir dessa mudança para um novo modelo de organização de produção, o mercado global de terceirização atingiu a marca de US$ 261.9 bilhões em 2022 e tem margem de crescimento anual com taxa de 9,4% até 2030 (Outsource Accelerator, 2023).

Com o alto número de organizações aderindo a modalidade interorganizacional, diversas companhias passaram a identificar a complexidade no fluxo de processos das vias de produção interoperantes. Nesse sentido, o investimento em mineração de processos tem crescido ano após ano a fim de identificar melhorias de processo no fluxo de negócio das organizações (Process Excellence Network, 2024). Algumas pesquisas, como a do quadrante mágico[1] da Gartner[2], apontam um aumento dos investimentos em mineração de processos de 39,5% em 2024 comparado ao ano de 2023, estimando uma média de aumento de 18% anualmente em uma projeção de 5 anos, saindo dos US$ 871,6 milhões registrados em 2023 podendo chegar a uma marca de US$ 2 bilhões até 2028 (Process Excellence Network, 2024).

A junção desse novo modelo de mercado e o interesse global em identificar cada vez mais os fluxos de trabalho dos processos interorganizacionais impulsionaram diversas pesquisas e projeções de mercado, e também pesquisas de setores econômicos como a projeção de mercado do DataBridge Market Research (2024), de que o mercado global de mineração de processos foi avaliado em US$ 324 milhões em 2023 e está projetado para atingir US$ 6.436,8 milhões até 2031, com uma CAGR[3] de 45,30% de 2024 a 2031. Já para MarketsandMarkets (2023), o mercado deve atingir um valor de US$ 12,1 bilhões até 2028, sendo US$ 1,8 bilhões em 2023, com uma CAGR de 45,6%. Os dados mostram as diferenças nas projeções e reforçam o dinamismo do mercado. Independente dos valores exatos das projeções, todas demonstram um crescimento significativo.

Diante deste cenário, é possível observar a importância de investigar a complexidade

[1] Metodologia de pesquisa que avalia e posiciona provedores de tecnologia em um mercado específico com base em dois critérios principais: Visão Abrangente e Capacidade de Execução. Ela organiza os fornecedores em quatro quadrantes: Líderes, Desafiadores, Visionários e Participantes de Nicho.

[2] Empresa global de pesquisa e consultoria, focada em tecnologia e serviços empresariais. Fornece informações e *insights* para ajudar executivos e suas equipes a tomar decisões estratégicas.

[3] Taxa de Crescimento Anual Composta

dos fluxos de trabalho interorganizacionais. Esse tipo de fluxo pode ser representado pela rede de Petri, que é um formalismo matemático que permite a construção gráfica de grafos direcionados eficaz na modelagem de sistemas concorrentes, assíncronos, distribuídos, paralelos, não determinísticos e estocásticos (MURATA, 1989). Uma rede de Petri que representa um fluxo de trabalho é denominada *WorkFlow net* ou WF-*net* (AALST; HEE, 2004). Portanto, uma WF-*net* é uma rede de Petri adaptada para modelar fluxos de trabalho em uma organização.

Contudo, para representar um fluxo de trabalho interorganizacional entre duas ou mais organizações utiliza-se uma extensão das WF-*net*, mantendo-se o mesmo princípio de modelagem. Neste caso, são necessárias duas ou mais WF-*net* locais para compor a representação formal (AALST, 1999). Este é o ponto onde as WF-*net* interorganizacionais (ou IOWF-*net*) diferem da definição padrão das WF-*net* locais, pois pontos de comunicação realizam a intercomunicação das redes envolvidas (AALST, 1999).

Com base nesses conhecimentos, diversos estudos acerca da mineração de processos foram iniciados na década de 1990 a partir principalmente do professor Wil Van Der Aalst[4], que foi um dos pioneiros nesta área de pesquisa. A mineração de processos permite descobrir, monitorar e melhorar processos reais através de informações coletadas a partir de diversos sistemas de informação (AALST, 2016).

O trabalho de Medeiros e Günther (2005), adaptou uma rede de Petri, modelada na ferramenta de modelagem CPN Tools, para gerar *event logs*. Segundo Aalst et al. (2003), *event logs* são um conjunto de registros de eventos onde cada evento equivale à representação de uma tarefa específica que simula um passo bem definido de um processo de negócio. Medeiros e Günther (2005) apresentaram um passo importante para que seja possível minerar a partir de simulações em modelos baseados em rede de Petri.

Já os trabalhos de Engel et al. (2016) e Aouachria et al. (2024), apresentaram a mineração de processo em situações do mundo real, utilizando dados de organizações reais, e realizaram a modelagem utilizando o formalismo das redes de Petri. Os dois trabalhos destacam que a IOWF-*net* é complexa e que o formalismo adotado em suas pesquisas é aplicável apenas aos casos analisados. Dessa forma, a modelagem de um novo cenário requer a reconstrução completa do modelo, com a aplicação total do formalismo matemático desde o início.

Diante desse contexto, essa pesquisa possibilitou a adaptação das *WorkFlow nets* interorganizacionais para a geração de *event logs*. Assim, foi possível contribuir para a superação de uma das limitações observadas na literatura sobre mineração de processos a partir de modelos WF-*net* e IOWF-*net*. Os estudos encontrados baseiam-se predominantemente em definições matemáticas formais para a conversão dos processos de negócio

[4] Professor Wil Van Der Aalst é considerado um dos pioneiros na área de mineração de processos. Página pessoal disponível em: <https://www.vdaalst.com/>. Acesso em: 4 jun. 2025.

das organizações, o que demanda alto domínio dos fundamentos matemáticos envolvidos. Portanto, essa pesquisa se destaca ao tornar esse processo mais acessível, reduzindo a complexidade exigida e promovendo maior aplicabilidade prática.

Sendo assim, o presente trabalho tem como objetivo apresentar um mecanismo de análise baseado na aplicação combinada de redes de Petri e mineração de processos para simulação e análise de processos interorganizacionais. Para tanto, especificamente, objetivou-se:

- definir um processo interorganizacional, representando as atividades, as entidades colaboradoras e seus fluxos de interação.
- modelar o processo definido, por meio das redes de Petri.
- estabelecer um mecanismo para a geração de *event logs* no modelo desenvolvido.

Posto isto, este trabalho está organizado da seguinte forma:

- Capítulo 2: apresenta o referencial teórico desta pesquisa, onde os principais conceitos para o entendimento completo do material abordado estão divididos. Tais quais rede de Petri, *WorkFlow net*, *WorkFlow net* interorganizacional, rede de Petri Colorida e mineração de processos.
- Capítulo 3: localiza-se os trabalhos relacionados que apresentam parte do conhecimento utilizado como base para esta pesquisa, ou que apresentam conhecimentos importantes para a realização dos objetivos desta pesquisa.
- Capítulo 4: onde encontra-se as etapas de escolha, construção e adaptação do modelo interorganizacional utilizado no mecanismo proposto.
- Capítulo 5: destina-se a apresentar as ferramentas utilizadas para realizar a mineração de processos e a análise dos resultados obtidos.
- Capítulo 6: demonstra os resultados obtidos com a pesquisa, e apresenta os trabalhos futuros.
- por fim se encontram as referências bibliográficas dos trabalhos utilizados no decorrer desta pesquisa.

2 Referencial Teórico

Este capítulo abordará conceitos e definições relevantes necessários para entender o mecanismo proposto nesta pesquisa. Para isso a abordagem de assuntos como redes de Petri e mineração de processos, presentes na seção 2.1 e seção 2.2, respectivamente, farão a composição deste capítulo.

2.1 Redes de Petri

As primeiras aparições do termo redes de Petri datam do início dos anos 60, mais especificamente em 1962, com a publicação da tese de Carl Adam Petri, na Faculdade de Matemática e Física da Universidade Técnica de Darmstadt, Alemanha Ocidental (CARDOSO; VALETTE, 1997). Sua dissertação foi bem aceita no campo de pesquisa despertando o interesse de grandes centros de pesquisa como o Applied Data Research, Inc. nos Estados Unidos, e do Projeto MAC de 1970 sobre sistemas simultâneos e computação paralela (MURATA, 1989).

Todo o reconhecimento e interesse por redes de Petri ocorreu por conta dos benefícios que contemplam a utilização dessa ferramenta, comportando uma aplicabilidade excelente em sistemas de processamento de informação que se caracterizam por serem concorrentes, assíncronos, distribuídos, paralelos, não determinísticos e/ou estocásticos. Além de conter elementos de comunicação visual semelhante a fluxogramas, diagramas de blocos e redes (MURATA, 1989).

Segundo Cardoso e Valette (1997), pode-se dizer que a rede de Petri é composta principalmente por 3 elementos básicos distintos, sendo eles (1) Lugar, (2) Transição e (3) Ficha que formam os primeiros conjuntos de regras. Esses elementos podem ser definidos da seguinte forma:

1. **Lugar**: um lugar geralmente corresponde a um predicado indicando um recurso, um procedimento, uma decisão, um estado parcial, etc. Graficamente um lugar é representado por um círculo, pelo qual estabelece a comunicação através de arcos com as transições.

2. **Transição**: a transição é representada graficamente por uma barra ou retângulo, sendo responsável por representar as operações e atividades de um sistema. Em geral, elas representam os eventos que ocorrem em um sistema.

3. **Ficha**: as fichas atuam como os recursos que serão utilizados no sistema, e geralmente são associadas aos lugares da rede. Uma ficha em um lugar representa que o predicado

associado aquele lugar é verdadeiro. Graficamente as fichas são representadas por pontos.

A Figura 1 ilustra os elementos básicos de uma rede de Petri.

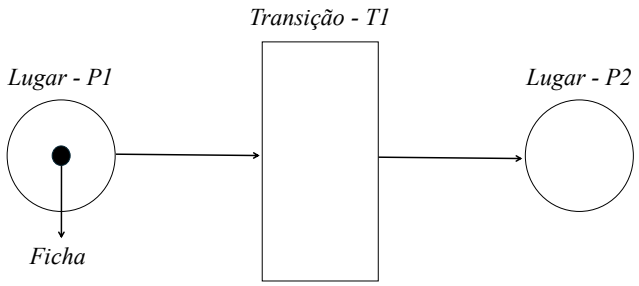


Figura 1 – Elementos básicos da rede de Petri.

Definição 1. *De acordo com Cardoso e Valette (1997), uma rede de Petri é uma quádrupla*

$$R = <P,\ T,\ Pre,\ Post>$$

onde:

- *P é um conjunto finito de lugares de dimensão n;*
- *T é um conjunto finito de transições de dimensão m;*
- *Pre : P × T → IN é a aplicação de entrada (lugares precedentes ou incidência anterior), com IN sendo o conjunto dos números naturais;*
- *Post : P × T → IN é a aplicação de saída (lugares seguintes ou incidência posterior).*

Na Figura 1, é possível ver os itens descritos na definição apresentada por Cardoso e Valette (1997). Na Figura 1, o Lugar - P1 contém uma ficha, dessa forma a Transição - T1 fica sensibilizada, indicando que é possível ser disparada, e o arco que estabelece a ligação entre o Lugar - P1 e a Transição - T1 ,representa o arco de entrada da transição, e transmite a ficha do lugar para a transição. Os arcos numa rede de Petri são ponderados. Os pesos nos arcos indicam quantas fichas são necessárias para disparar a transição. No caso da Figura 1, não há inscrição no arco, neste caso, considera-se o peso como sendo 1. Após a Transição - T1 ser sensibilizada a ficha antes presente no Lugar - P1 é consumida, e o arco que liga a Transição - T1 ao Lugar - P2 atua como arco de saída, assim o Lugar - P2 receberá uma ficha.

O exemplo ilustrado na Figura 2, apresenta um sistema da vida real, uma modelagem que representa o funcionamento de um semáforo. Neste exemplo os lugares atuam como os elementos físicos do semáforo representando as cores das lâmpadas do equipamento. Já as transições executam o papel dos eventos responsáveis por realizar a mudança do estado do semáforo, trocando a cor das lâmpadas. A ficha presente no lugar inicial representa o estado atual do semáforo, portanto, quando o lugar estiver marcado significa que a luz da cor referente aquele lugar está ativa.

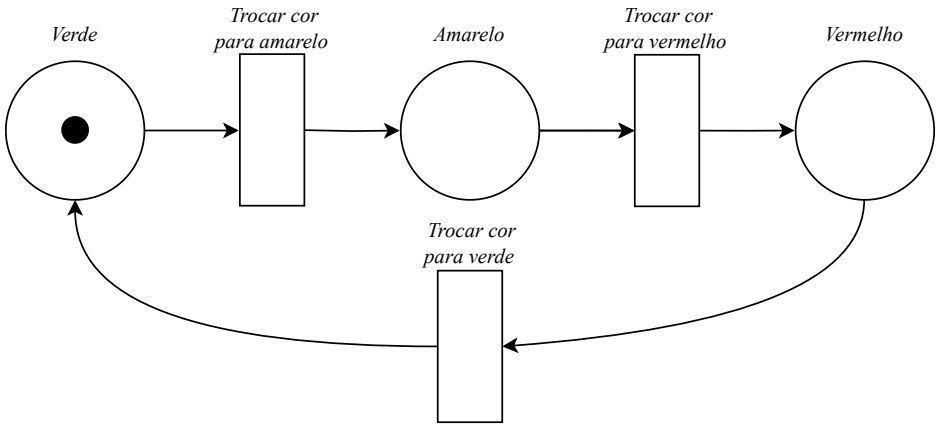
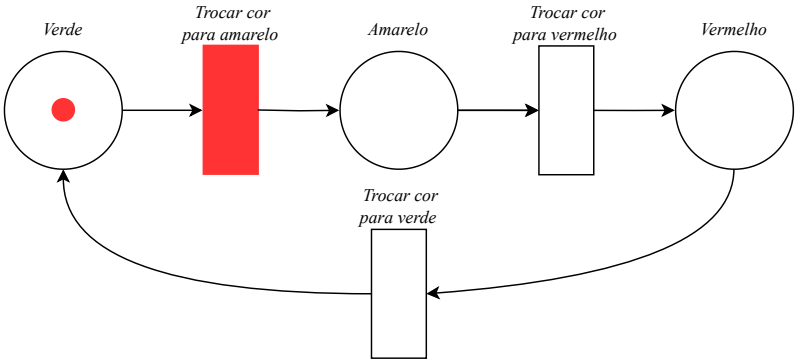


Figura 2 – Modelo de funcionamento de um semáforo em rede de Petri.

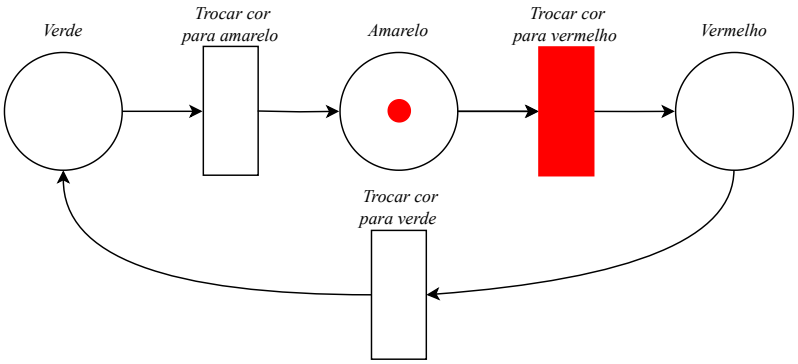
A Figura 3a ilustra o modelo do semáforo quando a transição Trocar cor para amarelo está sensibilizada. Após disparar, a ficha é depositada no lugar Amarelo, indicando que a cor mudou. A ficha neste lugar sensibiliza a transição Trocar cor para vermelho (Figura 3b). Por fim, a Figura 3c ilustra a ficha no lugar Vermelho e a transição Trocar cor para verde sensibilizada. Após o disparo dessa transição, a rede volta ao seu estado inicial e o ciclo se repete.

No exemplo da Figura 3 é possível perceber que após um disparo de uma transição, a ficha no lugar de entrada da transição é consumida e uma ficha é depositada em seu lugar de saída. Uma vez que não há fichas no lugar de entrada, a transição não pode mais ser disparada. Durante a execução do modelo, visualiza-se a troca de um estado discreto para outro, conforme Murata (1989) e Cardoso e Valette (1997) afirmam como sendo uma das características principais das redes de Petri.

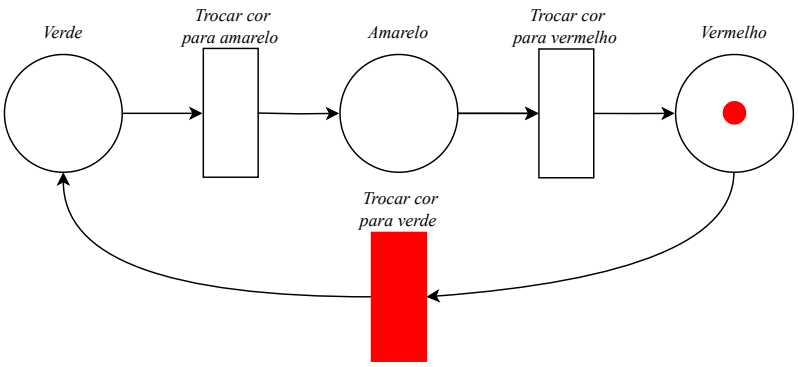
Uma das características abordadas por Cardoso e Valette (1997) está relacionada ao polimorfismo presente na representação das modelagens de rede de Petri, um exemplo é a capacidade dos elementos da rede de simular o comportamento de unidades de diversas magnitudes. Como no exemplo da Figura 3, a ficha foi utilizada para representar qual o



(a) Muda para o sinal amarelo.



(b) Muda para o sinal vermelho.



(c) Muda para o sinal Verde.

Figura 3 – Funcionamento da rede que representa um semáforo.

estado de um semáforo, mas ela também pode ser utilizada para representação de recursos e ativos. Além disso, os autores afirmam que é possível validar e verificar o modelo de acordo com a dinâmica de sua marcação, ou seja, de acordo com o disparo das transições e mudanças da ficha.

2.1.1 WorkFlow nets

De acordo com Aalst, Verbeek e Hauschildt (1997) e Aalst e Stahl (2011), um *WorkFlow* representa um fluxo de trabalho pertencente a organizações empresariais. O fluxo é composto por tarefas que, por sua vez, compõem os processos internos de cada organização. Apesar de representar o fluxo de trabalho, também é chamado muito comumente de processo de negócio. Uma rede de Petri que representa um processo de negócio é chamada de *WorkFlow net* (WF-*net*) (AALST; HEE, 2004).

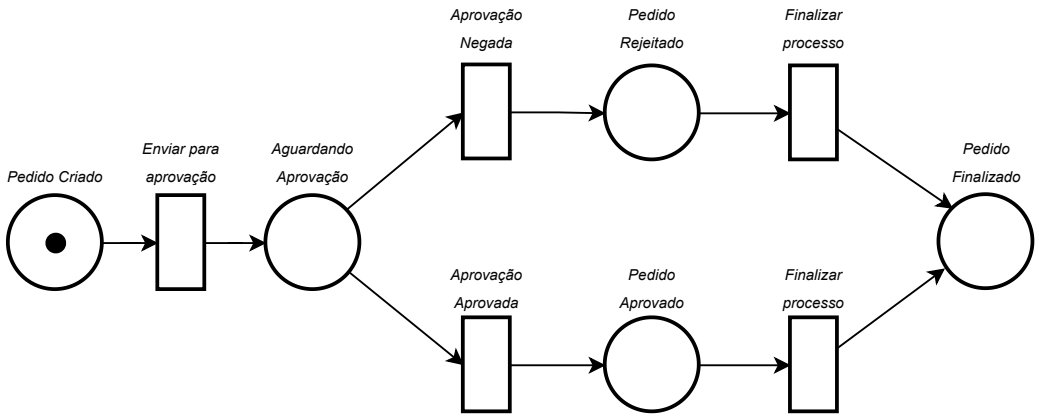


Figura 4 – Exemplo de um processo de negócio modelado por uma *WorkFlow net*.

A Figura 4 apresenta uma WF-*net* que modela o processo de um pedido *on-line*. Na ilustração, observa-se que o lugar com nome *Pedido Criado* é o estado de origem (ou estado inicial) *i*, já o lugar *Pedido Finalizado* representa o estado destino (ou estado final) *o*. Cada transição do modelo representa uma tarefa do processo de negócio.

Segundo Aalst e Hee (2004), alguns conceitos acerca de processos de negócios englobam certos elementos como:

- Caso: um caso pode ser interpretado como um produto, ou um serviço que ao final do processo de negócio deve ser finalizado. O intuito é de produzir ou modificar

algo, que tenha início e fim bem especificado, ou seja um ciclo de vida limitado. Esse "produto", ao final, de forma abstrata é o que se chama de caso.

- Tarefa: a tarefa é um elemento atômico do processo, representando uma ação que precisa ser realizada. A execução da tarefa pode ser feita por meio de uma sequência de etapas definidas pelo executor em um intervalo de tempo.
- Processo: define como uma categoria de casos deve ser executada e é responsável por indicar quais conjuntos de tarefas devem ser executados e a ordem em que devem ocorrer. O ciclo de vida de um caso também é definido por processos, portanto, assim como um caso, um processo também tem um início e fim bem definidos.
- Recurso: os recursos podem ser identificados como qualquer indivíduo capaz de executar uma tarefa, podendo ser uma equipe humana, robôs, um computador, uma impressora, etc.
- Roteamento: a estrutura que faz a distribuição das tarefas que precisam ser executadas e a ordem que devem ser executadas. Dentre as formas que o roteamento faz a distribuição das tarefas se encontram:
    - Sequencial: o roteamento sequencial é a mais simples delas, fazendo a execução das tarefas uma após a outra, na maioria dos casos são tarefas que dependem umas das outras, ou seja, o resultado de uma é essencial para a resolução da próxima tarefa. A Figura 5 ilustra o roteamento sequencial em uma WF-*net*.
    - Paralela: consiste em duas ou mais tarefas sendo executadas simultaneamente, em qualquer ordem. A lógica utilizada consiste em uma bifurcação gerada a partir de uma divisão *AND* seguida de uma junção *AND* que sincronize as tarefas novamente. O roteamento paralelo pode ser visto na Figura 6.
    - Condicional: é uma forma de roteamento que escolhe uma entre duas ou mais tarefas. A forma que essa escolha é feita é dependente das propriedades físicas do caso. O caminho utilizado para esse tipo de roteamento faz uso da divisão *OR*, e junção *OR* para os caminhos alternativos. A Figura 7 representa o roteamento condicional em uma WF-*net*.
    - Iterativa: é utilizada quando uma única execução da tarefa não é o suficiente para atender a verificação do caso. Nessa situação a tarefa é executada até que atenda a demanda subsequente. A iteração de uma tarefa pode ser vista na rede da Figura 8.

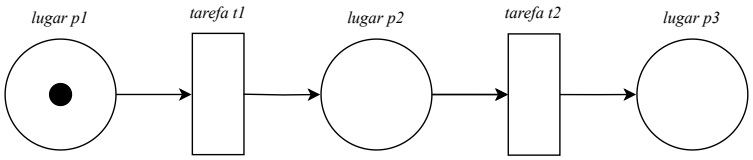


Figura 5 – Roteamento sequencial em uma WF-*net*.

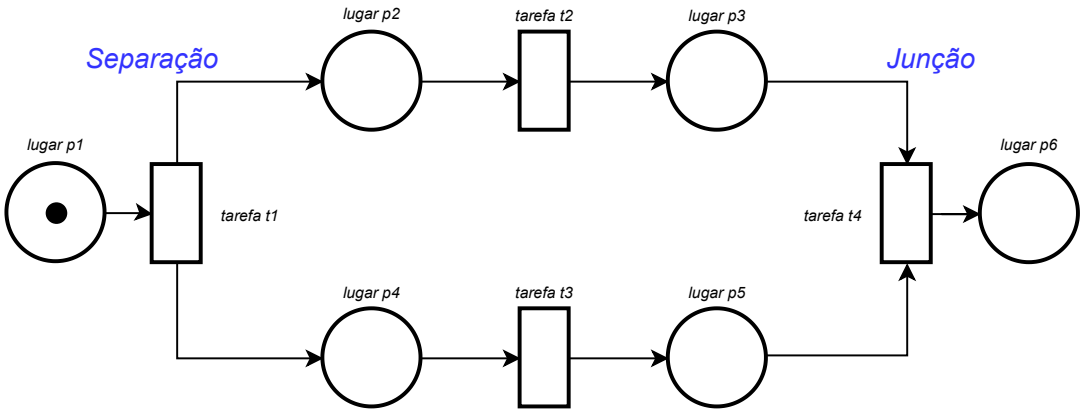


Figura 6 – Exemplo do roteamento paralelo em uma WF-*net*.

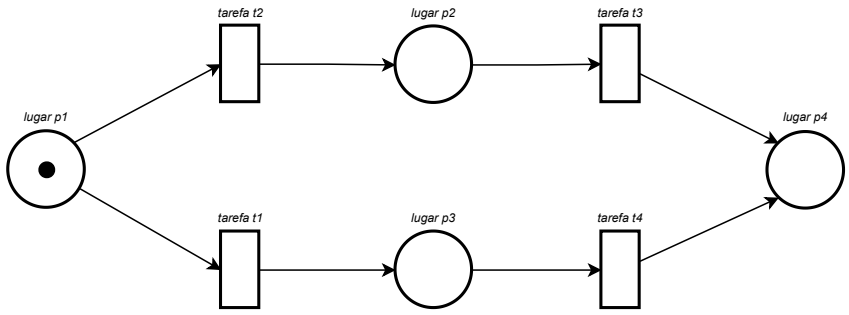


Figura 7 – Exemplo de um roteamento condicional em uma WF-*net*.

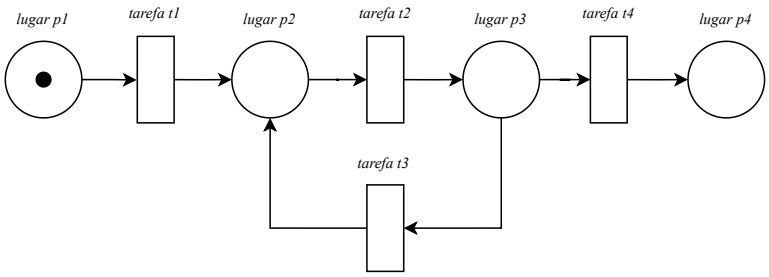


Figura 8 – Exemplo de um roteamento iterativo em uma WF-*net*.

Definição 2. *Uma Petri net PN = (P,T,F) é uma WF-net (WorkFlow net) se e somente se (AALST; HEE, 2004):*

- *Há um local de origem $i \in P$ tal que $\bullet i = \emptyset$.*
- *Há um local de destino $o \in P$ tal que $o\bullet = \emptyset$.*
- *Todo nó $x \in P \cup T$ está em um caminho de i para o.*

O primeiro item a definição estabelece que o lugar de origem é um lugar composto apenas por arcos de saída. Já o segundo item diz que o lugar de destino é um lugar composto apenas por arcos de entrada. O terceiro item da definição estabelece uma relação

de conexão entre cada nó da WF-*net* para que não haja pontas soltas na rede, mantendo assim, todas as tarefas dos processos entre os lugares de origem e de destino. (AALST; HEE, 2004).

Segundo Aalst e Hee (2004), na WF-*net* os lugares representam as condições, as transições representam as tarefas as fichas representam os casos. Aalst (1996) e Aalst e Stahl (2011) abordam uma das principais propriedades da WorkFlow net, responsável por garantir a corretude da rede, chamada de propriedade *Soundness*. Para isso, a WF-*net* deve satisfazer alguns critérios para que esteja de acordo com os requisitos da propriedade, como:

- Opção de completar *(Option to complete)*: onde para cada estado p a partir do estado de origem i, através de uma determinada quantidade de disparos é possível alcançar o estado de destino o, dessa forma assume-se que todo caso terá um término.
- Conclusão adequada *(Proper completion)*: quando uma ficha chegar ao estado de destino todos os outros estados a partir da origem até o destino devem estar vazios.
- Ausência de tarefas mortas *(Absence of dead activties/transitions)*: significa que ao final do processo, todas as tarefas foram executadas. Em termos de WF-*net*, isso significa que a partir da marcação inicial, todas as transições da rede podem ser disparadas.

A validação da propriedade *Soundness* foi apresentada e provada nos trabalhos de (AALST, 1996), (AALST, 1997) e (AALST, 1998). Para que ela seja validada, a WF-*net* deve passar por uma leve adaptação, onde o lugar de origem e o lugar de destino são ligados através de uma transição $t*$. A Figura 9 ilustra a rede de Petri com a transição $t*$. Após essa adaptação, a rede modificada é analisada e verifica se os três requisitos citados anteriormente são atendidos. Esses requisitos podem ser verificados por meio da análise de propriedades bem conhecidas das redes de Petri, a *liveness* (vivacidade) e *boundedness* (limitabilidade):

- *Liveness*: uma rede de Petri é considerada *live* (viva) se, para todo estado alcançável M' e toda transição t, existe um estado M" alcançável a partir de M' que habilita t (AALST; HEE, 2004). Em outras palavras, não importa o estado em que a rede esteja (desde que seja um estado alcançável), sempre deve ser possível, através de alguma sequência de disparos, chegar a um estado onde qualquer transição desejada possa eventualmente disparar. Uma rede de Petri (ou WF-*net*) é considerada *live* se todas as suas transições são *live*.
- *Boundedness*: uma rede de Petri é limitada *(bounded)* se, e somente se, para cada lugar existir um número natural n tal que, para todo estado alcançável o número de

fichas é menor ou igual a *n*. A rede é segura *(safe)* se, e somente se, para cada lugar o número máximo de fichas não exceder 1 (AALST; HEE, 2004). Isso significa que o número de fichas em qualquer lugar nunca cresce indefinidamente.

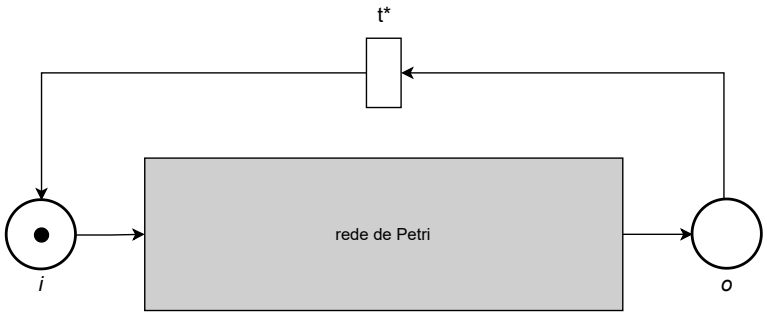


Figura 9 – Exemplo de uma WF-*net* adaptada para a verificação da propriedades *Soundness*.

2.1.2 WorkFlow net interorganizacional

De acordo com Aalst (1999), uma *WorkFlow net* interorganizacional reúne de forma consistente os mesmos conceitos de uma WF-*net*, porém, os modelos interorganizacionais, trazem complementos adjacentes às organizações presentes na modelagem, pois se trata da modelagem de processos de negócios entre duas ou mais organizações que possuem fluxos de trabalho independentes. A abordagem da representação da *WorkFlow net* interorganizacional segue a interoperabilidade *loosely coupled* (fracamente acoplados) reforçando a característica de independência de processos.

Uma das características da WF-*net* interorganizacional (IOWF-*net*) é ser composta por duas ou mais WF-*net* locais. A junção de duas ou mais WF-*nets* através de pontos de comunicação regem a formação da IOWF-*net*. Segundo Aalst (1999), as comunicações entre as WF-*nets* podem ocorrer de duas maneiras, síncrona e assíncrona. A comunicação síncrona estabelece a execução de tarefas específicas de maneira simultânea, já a comunicação assíncrona é voltada para a troca de mensagens entre as redes.

Justamente por se tratar da junção de diferentes fluxos de trabalho, processos, especificações, e até mesmo dados diferentes, os modelos de IOWF-*net* são potencialmente complexos. Essa característica torna a validação da rede complexa também sendo necessário adaptações no modelo para validar a propriedade *Soundness*. Aalst (1999) afirma que uma adaptação significativa deve ser aplicada na implementação da análise *Sound* de

uma IOWF-*net* em comparação com as WF-*net* locais. Essa adaptação consiste em duas análises, local e global. A análise local realiza uma varredura em cada uma das WF-*net* locais da IOWF-*net* global. Já a análise global analisa um modelo adaptado da IOWF-*net* onde uma nova modelagem é feita com a junção de todas as WF-*net* locais em apenas um modelo global, esse novo modelo é chamado de rede desdobrada *(Unfolded net)*. Somente com essas duas varreduras é possível dizer se a IOWF-*net* é *Sound* ou não.

A Figura 10 representa uma IOWF-*net* (AALST, 1999) que é *Sound* apenas localmente. Observa-se que o modelo agrupa duas redes locais denominadas *LWF1* e *LWF2* que são iniciadas a partir dos lugares de origem *i1* e *i2*, e se estendem até os lugares de destino *o1* e *o2*. Outros elementos importantes da modelagem são os lugares de comunicação assíncrona *CA1, CA2 e CA3*, e comunicação síncrona *CS1*. Esse exemplo destaca um problema na adaptação do modelo local para uma IOWF-*net* onde ocorre um erro na comunicação assíncrona *CA2*, ocasionado por uma ficha que ficaria "presa" caso as transições *T4 e T12* sejam disparadas, dessa forma o lugar de comunicação assíncrona *CA2* teria uma ficha "morta".

Já a Figura 11 é um exemplo de IOWF-*net* Aalst (1999) que está correta apenas globalmente, ou seja somente a modelagem da rede desdobrada é *Soundness*, isso significa que uma, ou ambas as redes locais não estão corretas, neste caso a rede local denominada *LWF2* não está correta, pois as fichas podem ficar paradas no lugar *P14*.

Por fim, a Figura 12 apresenta o modelo de uma IOWF-*net* Aalst (1999) *Soundness*. Se uma análise local for aplicada em cada uma das WF-*net* locais (*LWF1 e LWF2*), ambas irão atender aos requisitos necessários da propriedade *Soundness*. O mesmo vale para a análise da rede desdobrada, pois ela também atende às condições da propriedade *Soundness*.

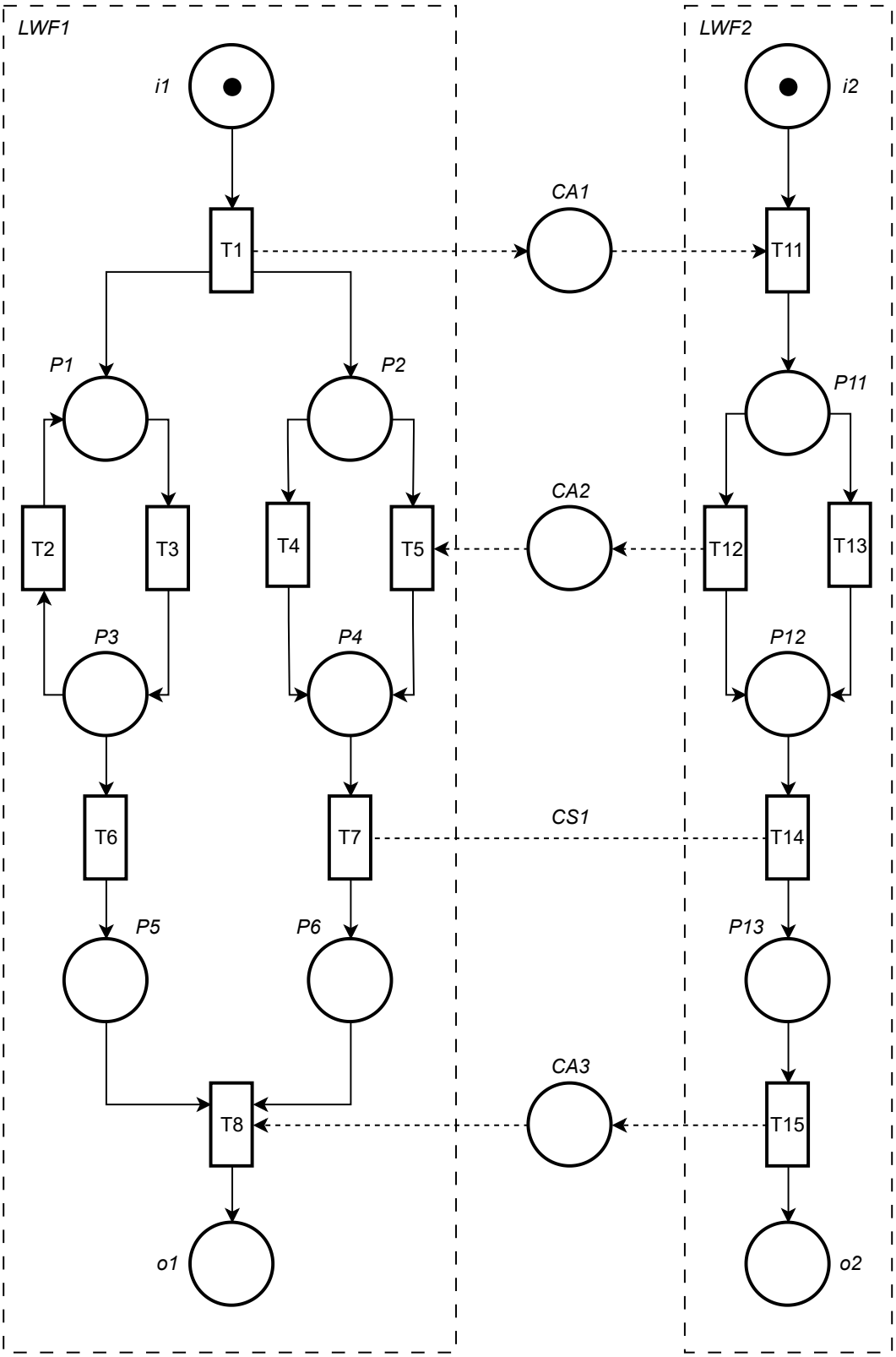


Figura 10 – Exemplo de uma IOWF-*net Sound* local.

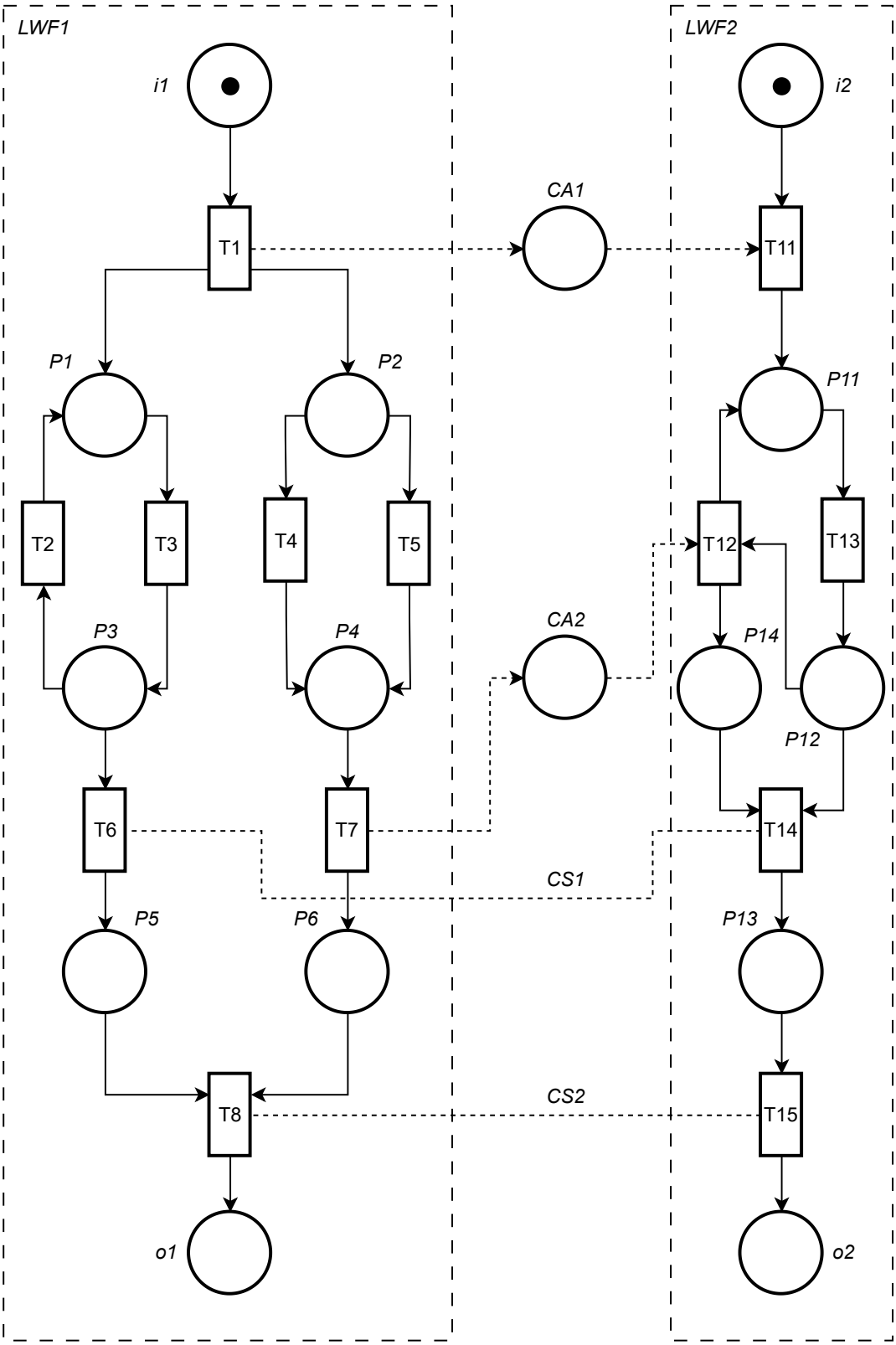


Figura 11 – Exemplo de uma IOWF-*net Sound* global.

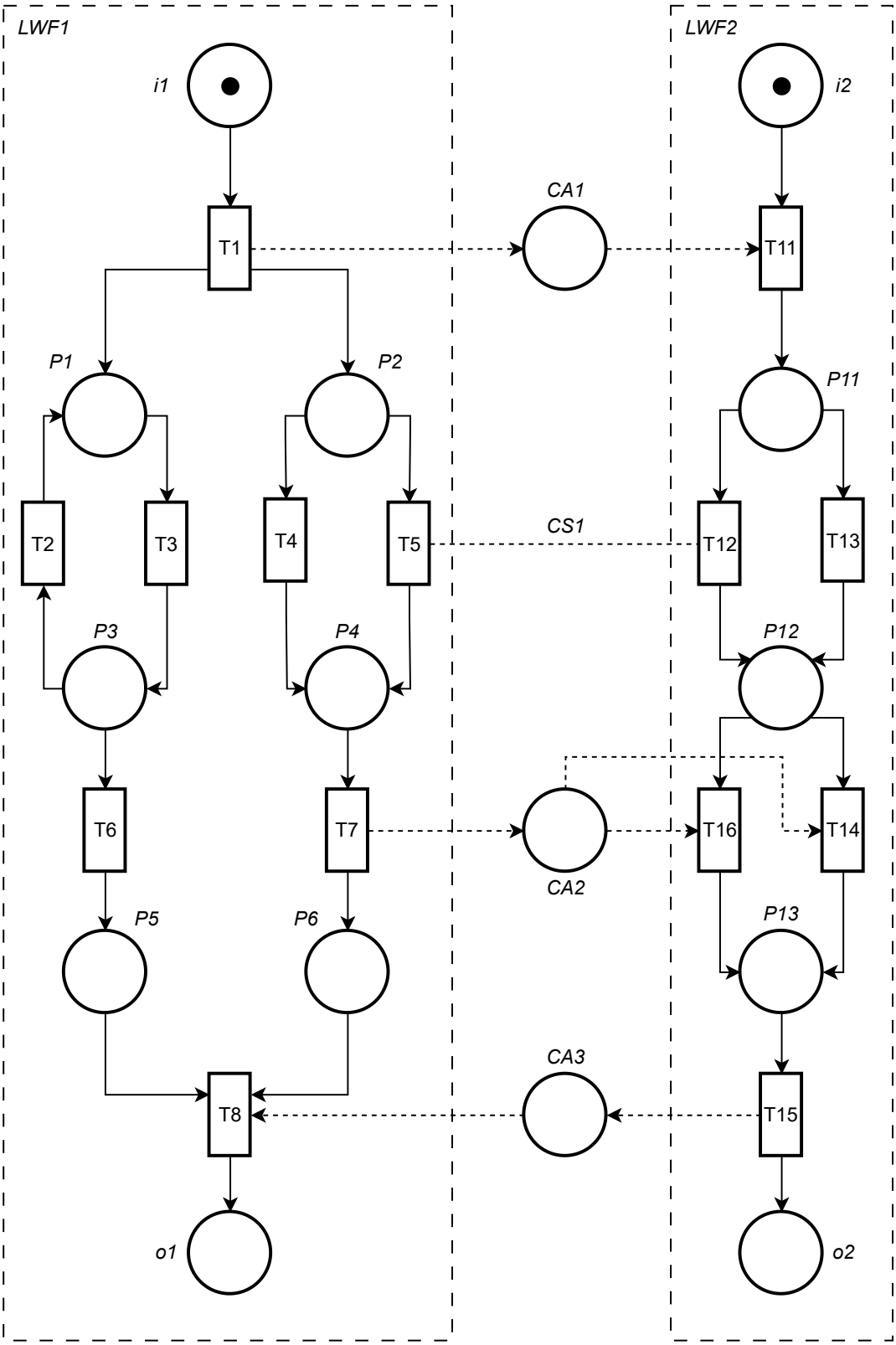


Figura 12 – Exemplo de uma IOWF-*net Sound*.

2.1.3 Redes de Petri Coloridas

Segundo Jensen e Kristensen (2009), as redes de Petri Coloridas (CPN ou CP-*nets*), são uma linguagem gráfica para modelagem e construção de sistemas concorrentes a fim de realizar a verificação das propriedades destes sistemas. Jensen e Kristensen (2009) afirmam que as CP-*nets* são uma linguagem de modelagem de eventos discretos, usando a sua capacidade em conjunto com uma linguagem de programação de alto nível, chamada CPN ML, baseada na linguagem Standard ML com paradigma funcional (JENSEN, 1996).

Sua capacidade em modelar sistemas com características onde a separação de dados deve ser feita usando tipos de dados abstratos ou complexos é um dos principais motivos da sua utilização. Jensen (1996) e Jensen e Kristensen (2009), mencionam que muitas vezes a rede de Petri ordinária cresce excessivamente em tamanho ou até mesmo gera muitos erros ao tentar representar tipos de dados complexos. De acordo com Jensen (1996), isso acontece pois a única maneira de diferenciar os *tokens* seria agrupando-os em lugares diferentes. Essa abordagem em sistemas complexos cria então modelos muito grandes e complexos para se manipular. Com as CPNs esse problema pode ser contornado. A capacidade da CP-*net* em conjunto com a programação de alto nível, consegue representar em um único modelo várias instâncias com tipos de dados diferentes, gerando redes menores e com fluxos mais dinâmicos.

Portanto, um dos principais pontos de diferença da rede de Petri Colorida está na forma como as fichas são definidas. As cores que podem ser adicionadas as fichas representam os tipos de dados definidos pelo usuário que está criando a rede. Dessa forma a informação é aplicada diretamente na ficha. Os tipos inseridos nas fichas podem ser tratados através de funções booleanas, denominadas funções de guarda, que são inseridas diretamente nas transições. As funções de guarda são responsáveis por habilitar ou não as transições. Os arcos da rede, por sua vez, especificam expressões que podem ser desde uma variável a conjunto de variáveis ou funções complexas. Assim, os arcos de entrada em uma transição, especificam os valores que serão consumidos. Enquanto que os arcos de saída de uma transição especificam quais valores serão produzidos.

Definição 3. *Uma rede de Petri Colorida Jensen e Kristensen (2009) não hierárquica é uma nove-tupla*

$$CPN = (P,\ T,\ A,\ \Sigma,\ V,\ C,\ G,\ E,\ I)$$

onde:

1. *P é um conjunto finito de lugares.*
2. *T é o conjunto finito de transições T tal que* $P \cap T = \emptyset$.

3. $A \subseteq P \times T \cup T \times P$ *é o conjunto de arcos direcionados.*

4. Σ *é um conjunto finito de conjunto de cores (color sets) não vazios.*

5. V *é um conjunto finito de variáveis tipadas tal que* $Tipo[v] \in \Sigma$ *para toda variável* $v \in V$.

6. C: $P \rightarrow \Sigma$ *é a função do conjunto de cores que associa a cada lugar um conjunto de cor.*

7. G: $T \rightarrow EXPR_v$ *é uma função de guarda (expressão booleana) que associa uma guarda a cada transição* t *tal que* $tipo[G(t)] = Bool$.

8. E: $A \rightarrow EXPR_v$ *é uma função de expressão de arco que associa uma expressão a cada arco tal que* $Tipo[E(a)] = C(p)_{MS}$, *onde* p *é o lugar conectado ao arco* a.

9. I: $P \rightarrow EXPR_\emptyset$ *é uma função de inicialização que atribui uma expressão de inicialização a cada lugar* p *tal que* $Tipo[I(p)] = C(p)_{MS}$.

De acordo com Jensen e Kristensen (2009), cada lugar em uma CPN possui um conjunto de cores (tipo) associado, e a marcação de um lugar é o multiconjunto de fichas que ele contém, onde a cor (valor) de cada ficha deve pertencer ao conjunto de cores daquele lugar. O sistema é definido com uma marcação inicial M_0 distinta. As transições possuem variáveis livres que aparecem em suas guardas e expressões de arco. Para garantir a corretude do modelo, o simulador da CPN ML verifica a consistência de tipo de todas as inscrições, assegurando, por exemplo, que uma expressão de arco produza fichas do tipo correto para o lugar de destino, evitando erros de modelagem.

A Figura 13 representa um modelo simples de uma CP-*net* que ilustra as principais características da CPN. O modelo conta com 3 lugares denominados respectivamente *Solicitação Recebida, Em Análise e Aprovado/Rejeitado*, que estão ligados por arcos que conectam as 2 transições presentes na rede, denominados *Iniciar Análise e Finalizar Análise*. No lugar *Solicitação Recebida*, nota-se a presença das fichas e suas respectivas cores, com isso é possível perceber que o valor de cada ficha segue o padrão definido do lugar em que ela se encontra. Neste exemplo o tipo de dado dos lugares é o resultado de um produto cartesiano entre um inteiro *(INT)* e uma *string (STRING)*. As variáveis utilizadas no modelo seguem o mesmo padrão dos *color sets* dos lugares da modelagem. Uma característica importante deste modelo está nas transições *Iniciar Análise e Finalizar Análise*, pois a cada vez que a ficha sai da transição para o próximo lugar através dos arcos de saída, os valores são alterados, atualizando o resultado das variáveis da rede.

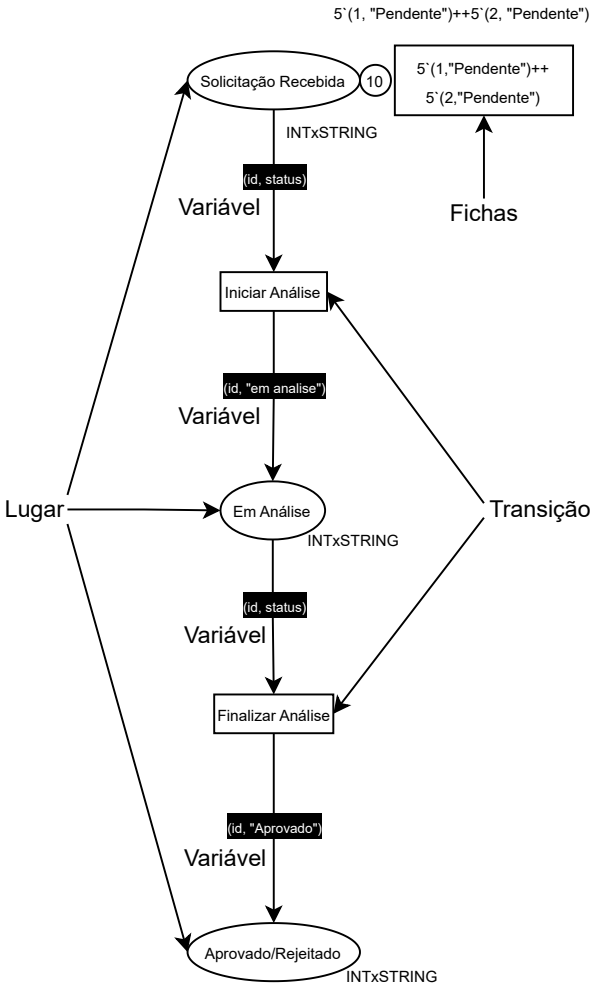


Figura 13 – Exemplo de uma rede de Petri Colorida.

2.1.4 CPN Tools

Para realizar a modelagem e aplicação de testes para a validação das redes é indicado o uso de *softwares* que facilitem a tarefa dos usuários. Para isso o CPN Tools é uma ferramenta capaz de realizar edição, simulação, análise de espaço de estados e análise de desempenho de modelos CPN (JENSEN; KRISTENSEN, 2009). O *software* possui capacidade de interpretar redes temporizadas, não temporizadas, hierárquicas e não hierárquicas, e está disponível para os principais sistemas operacionais, Windows e Linux.

A Figura 14 é um *screenshot* da área de trabalho do CPN Tools, com as ferramentas que fazem parte da aba *"Tool box"* no menu lateral localizado ao lado esquerdo da tela. Para que cada uma das ferramentas que compõem a imagem sejam abertas, é necessário

que o usuário clique com o botão esquerdo do mouse no nome da ferramenta escolhida, localizada no menu lateral, e arraste para a área de trabalho do CPN Tools. Feito o passo anterior, uma janela com os itens da ferramenta escolhida é aberta, esta janela tem o nome de *binder*. Cada *binder* na imagem representa uma ferramenta específica da aba de ferramentas, onde algumas delas são muito importantes.

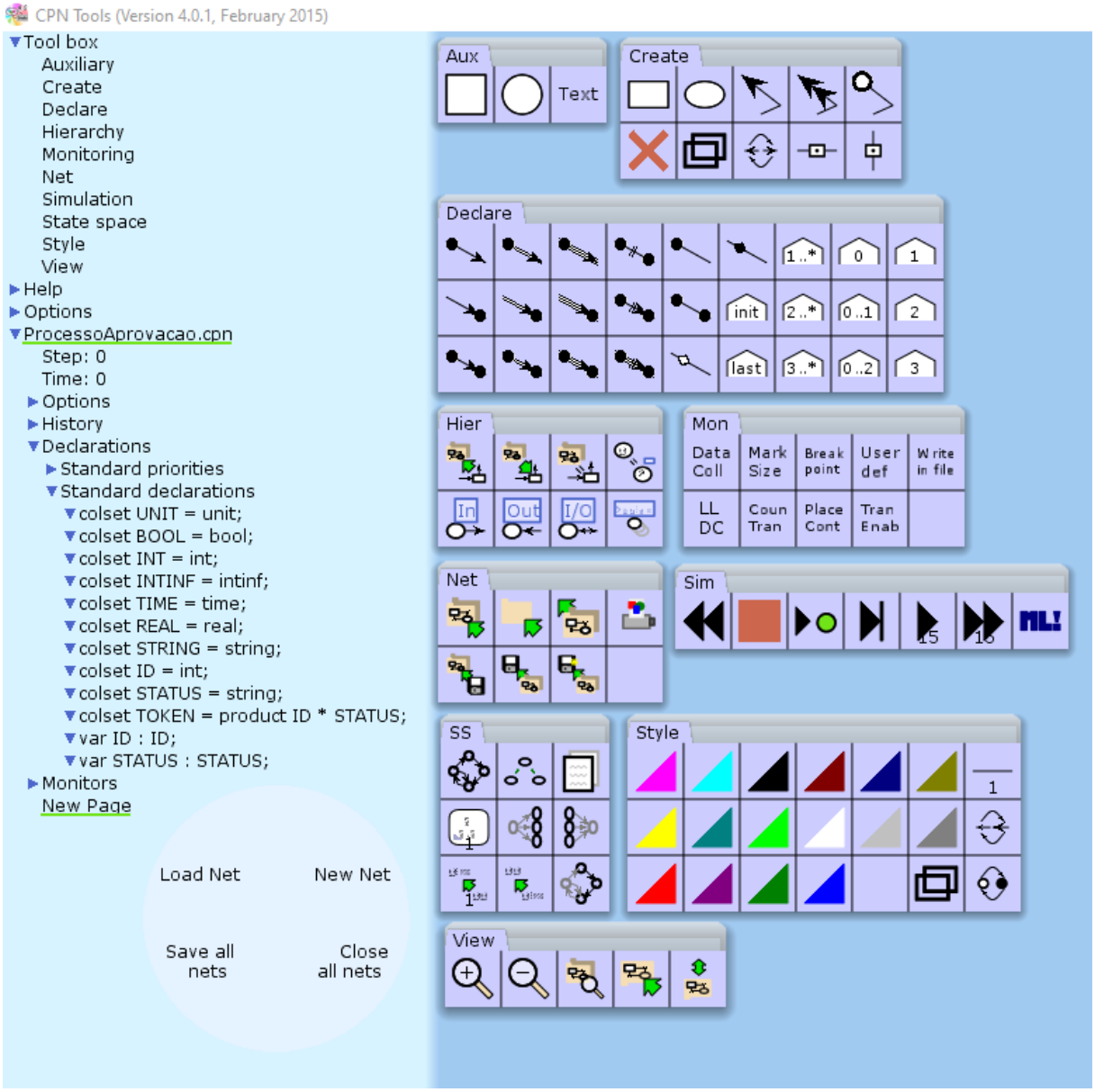


Figura 14 – *Screenshot* das ferramentas do CPN Tools.

Para a criação dos modelos pode-se utilizar os itens de criação do *binder "Create"*, e estilizá-los com funções do *binder "Style"*. Os conjuntos de *binders* mais utilizados no processo de criação e validação de um modelo geralmente são *"Create", "State Space", "Simulation" e "View"*, pois permitem respectivamente a inserção dos elementos da rede, gerar o relatório de estado de espaço, fazer simulações iterativas ou automáticas do modelo e navegar pelo *binder* que aloja o modelo da rede utilizando as opções de visualização.

No canto inferior esquerdo da Figura 14 pode-se observar um menu flutuante circular, este menu representa a opção que os usuários tem ao segurar com o botão direto do mouse sobre algum elemento do CPN Tools. A opção que é representada na imagem aparece ao segurar o botão direito do mouse em alguma área ociosa do CPN Tools, e com ela é possível carregar uma rede pronta, criar uma nova rede, salvar as redes atuais ou fechá-las. Esse menu flutuante circular fornece opções individuais para cada elemento do qual o usuário clicar e segurar o botão direito do mouse, revelando novas funcionalidades.

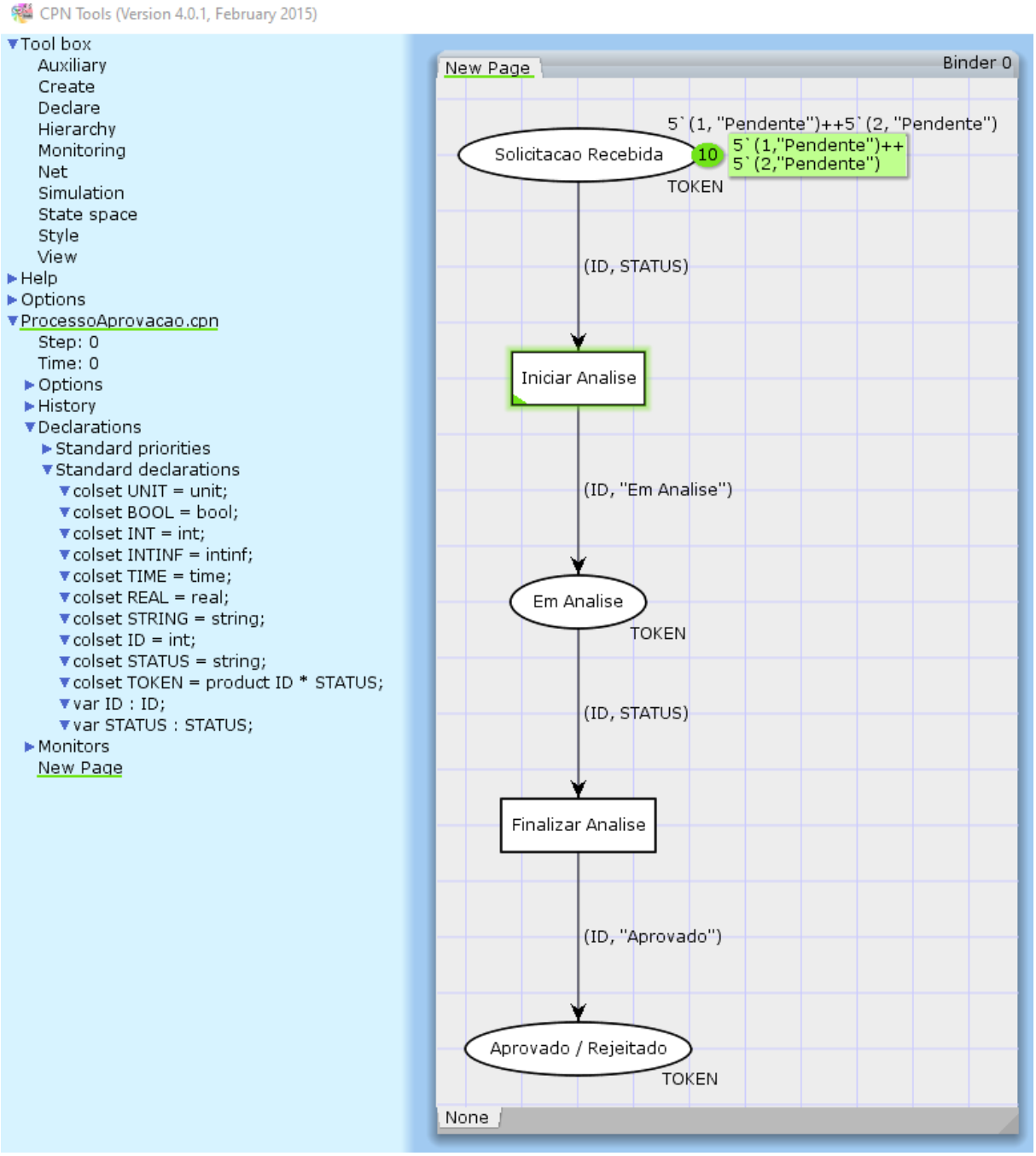


Figura 15 – *Screenshot* de um modelo criado no CPN Tools.

A Figura 15 apresenta um *screenshot* com um modelo básico de CP-*net*, nela a área de trabalho *"new page"* comporta a modelagem da rede, com todos os seus elementos. Na

Figura 15 também é possível notar o menu lateral da plataforma, específicamente a aba de *"Declarations"*, pois é nesta aba que *os color sets* são definidos. No submenu *Standard declarations* são definidos os *color sets* padrões e os que o usuário definiu para o modelo de rede em questão. Neste local também é onde são inseridas as variáveis de uma rede juntamente com as funções implementadas.

Ainda no menu lateral esquerdo encontra-se a aba de *Monitors* que são monitores programáveis, que são utilizados para monitorar a rede e realizar a extração de dados sobre o desempenho do sistema, como o tamanho de filas, atrasos e utilização de recursos. Os resultados podem ser exportados para relatórios estatísticos, incluindo médias, desvios padrão e intervalos de confiança (JENSEN; KRISTENSEN, 2009).

2.2 Mineração de Processos

De acordo com Aalst (2016) e Aalst et al. (2003), a mineração de processos é um campo de pesquisa ascendente, relativamente novo, que tem o intuito de descobrir, monitorar e melhorar processos reais através de informações coletadas a partir de diversos sistemas de informações como ERP (*Enterprise Resource Planning*[1]), CRM (*Customer Relationship Management*[2]), WFMF (*Workflow Management System*[3]), e BPM (*Business Process Management*[4]). Desta forma a mineração de processos executa um caminho inverso, utilizando os dados minerados para auxiliar na análise e projeto de processos de negócio.

Aalst, Weijters e Maruster (2004) afirmam que antes mesmo de começar um projeto de um processo de negócio, deve-se iniciar um estudo, analisando os processos e captando informações relevantes. O trabalho de Aalst et al. (2003) sustenta essa afirmação ao relatar que as modelagens de processos existentes sofrem muita interferência de percepções por parte do projetista, grande parte da modelagem ressalta como o fluxo de processos deveria ocorrer e não como realmente o processo se manifesta. Por conseguinte, as modelagens são representadas com excesso de subjetividade, portanto uma maneira de mitigar esse problema é usando os dados de processos obtidos através de análises de casos reais.

Para realizar a análise dos casos reais, segundo Aalst et al. (2003) e Aalst (2016), deve-se averiguar os registros de eventos *(event logs)*, que é um dos elementos mais importantes da mineração de processos, pois não são apenas dados em uma tabela, eles devem seguir um padrão, uma hierarquia e ter requisitos mínimos. Os *event logs* podem ser considerados o coração da mineração de processos, pois a partir da organização deles que podem ser mineradas as informações. Aalst (2016) afirma em seu trabalho que um

[1] Planejamento de Recursos Empresariais
[2] Gestão do Relacionamento com o Cliente
[3] Sistemas de Gerenciamento de *Workflow*
[4] Gerenciamento de Processos de Negócios

processo consiste em casos, um caso consiste em eventos. Quando se olha para dentro de um caso é possível observar eventos ordenados que podem ter atributos.

Quanto ao registro de cada evento, segundo Aalst et al. (2003), cada um corresponde a uma tarefa específica, com a tarefa representando um passo bem definido no processo, cada um desses eventos deve se referir a um caso que representa uma instância do processo e, por fim, todos os eventos devem ser bem ordenados. A Tabela 1 apresenta um modelo de *event log* com vários elementos e alguns dados em suas informações.

ID Caso	ID Evento	Timestamp	Tarefa	Originador	Tipo Evento
C10293	E12931	02/06/2025@08:15:12	Início do processo	Alice	start
C10293	E12932	02/06/2025@08:18:40	Validação de dados	Bob	complete
C10293	E12933	02/06/2025@10:45:05	Aprovação gerencial	João	complete
C10293	E12934	02/06/2025@10:50:10	Encerramento	Maria	complete
C20485	E20481	05/06/2025@07:59:33	Início do processo	Alice	start
C20485	E20482	05/06/2025@08:01:12	Validação de dados	Bob	complete
C20485	E20483	05/06/2025@07:50:00	Aprovação gerencial	João	complete
C20485	E20484	05/06/2025@08:20:45	Encerramento	Maria	complete
C30877	E30871	10/06/2025@13:45:00	Início do processo	Alice	start
C30877	E30872	10/06/2025@13:50:00	Validação de dados	Bob	complete
C30877	E30873	10/06/2025@14:10:00	Revisão técnica	Maria	complete
C30877	E30874	10/06/2025@14:45:00	Aprovação gerencial	Maria	complete
C30877	E30875	10/06/2025@15:00:00	Encerramento	Maria	complete

Tabela 1 – Exemplo de *log* com variações no processo (anomalias e desvios).

Na Tabela 1 encontram-se elementos importantes como o *ID* de cada caso, o *ID* de cada evento atrelado ao seu caso específico, um *Timestamp*, o nome da tarefa realizada em um evento, nome do originador (ator) da tarefa e o tipo do evento realizado. Todos esses elementos são componentes presentes em *event logs* para realizar a mineração de processos, mas há casos que terão uma quantidade inferior ou superior de dados. Porém salienta-se que o mínimo necessário apontado por Aalst (2016) para que haja a possibilidade de minerar algo, são as seguintes informações: ID do caso, nome da tarefa e uma ordem dos eventos que especifique sua ordem cronológica na falta de um *Timestamp*. Ao analisar a Tabela 1 pode-se identificar algumas anomalias como o gargalo de tempo da execução da tarefa *Aprovação gerencial* com ID *E12933*, pois o tempo para ela ser completada destoa do restante das mesmas tarefas do *log*, podendo indicar alguma atividade que pode ser evitada no futuro. Outra anomalia identificada está no evento com ID *E20483*, onde a ordem temporal dos eventos está incorreta, podendo indicar incongruências de registro ou até mesmo concorrência paralela. O evento com ID *E30873* representa uma tarefa extra em comparação do caso *C30877* com outros casos, indicando um desvio do processo padrão. Por fim, a última anomalia indicada pelo *Originador Maria* no evento com ID *E30874* realizando uma tarefa que não era designada para ela normalmente, indicando sobrecarga de trabalho, substituição de pessoal ou algum erro de registro.

A área de mineração de processos como um todo aborda diversas técnicas e procedimentos justamente por se tratar de um setor com um campo de aplicação muito extenso. Porém, existem três tipos de metodologias muito abordadas na literatura acerca de mineração de processos, sendo elas:

- ***Discovery:*** se trata do método utilizado para realizar a descoberta do modelo de um processo de negócio, usando apenas dados provenientes dos *event logs* sem utilizar nenhum tipo de informação *a priori*.
- ***Conformance:*** esta análise é feita utilizando uma espécie de comparação de conformidade entre os resultados de *event logs* em comparação ao modelo preestabelecido, do mesmo processo. Com isso é possível verificar se o modelo está de acordo com o esperado, pois os dados provenientes dos *event logs* revelam o comportamento real da aplicação. O trabalho de Rozinat e Aalst (2008) apresenta de forma detalhada e prática a implementação de um *framework* chamado *Conformance Checker* utilizado para analisar um caso real de uma prefeitura holandesa.
- ***Enhancement:*** já esta técnica é responsável por aplicar o conceito de alteração ou extensão do modelo do processo a partir de descobertas resultantes do *event log*. Operando em contraste ao método de *conformance*, realizando comparações entre o modelo e o registro de *logs* de eventos, este faz o incremento de novas partes ao modelo, ou simplesmente altera algum componente da modelagem.

A aplicação da mineração de processos não se limita apenas ao fluxo de atividades de uma organização. De acordo com Aalst (2016), também é possível descobrir e analisar outros elementos importantes a nível de gestão de pessoas, tempo e recursos. Portanto, os dados presentes nos *event logs* revelam diferentes perspectivas após a mineração de processo, as quais são:

- **A perspectiva de fluxo de controle:** essa perspectiva foca em canalizar os dados em prol da construção de todas as vias que os processos geram dentro de uma aplicação, ou seja, funciona como uma espécie de mapeamento dos possíveis caminhos que as atividades podem percorrer no fluxo de trabalho.
- **A perspectiva organizacional:** tem o intuito de descobrir fatos ocultados nos *event logs*, demonstrando o relacionamento entre as entidades de uma organização, por exemplo, os atores (pessoas, sistemas, departamentos, etc.). Com essa visão da perspectiva organizacional, os gestores podem tomar inciativas assertivas baseadas em evidências, replicando bons relacionamentos, e alterando más conexões de uma organização.

- **A perspectiva de caso:** além dos casos serem caracterizados pelas trajetórias que percorrem no processo e pelos recursos envolvidos, eles também podem conter informações relevantes em nível de seus atributos. Ou seja, é essencial rastrear e analisar os dados associados a cada caso individualmente, para entender seu comportamento específico dentro do processo.

- **A perspectiva de tempo:** concentra-se na análise dos aspectos temporais do processo, buscando identificar gargalos, atrasos, tempos de espera, subutilização de recursos e outras ineficiências relacionadas ao uso do tempo. Essa perspectiva depende da presença de *Timestamps* nos *event logs*, que permitem medir e correlacionar a duração e o intervalo entre as atividades ao longo do processo.

Após conhecer e entender como as técnicas de mineração funcionam, e compreender sobre suas diferentes perspectivas de execução, é importante assimilar a relação entre o modelo do processo e os *event logs*. Para isso, o trabalho de Aalst (2016) apresenta essa relação com os conceitos de *Play-Out, Play-In e Replay*, da seguinte maneira:

- ***Play-Out:*** o *Play-Out* representa a forma clássica de usar o modelo de processos, pode ser usado para analisar e executar um um processo de negócio. Aplicando uma ficha *"Player"* em uma rede de Petri repetidas vezes o comportamento da rede é revelado. Ou seja, o comportamento da rede é identificado de dentro para fora, seguindo as regras de execução do modelo em um ambiente simulado, funcionando analogamente a um fluxo de trabalho.

- ***Play-In:*** já o *Play-In* tem funcionamento inverso ao procedimento do *Play-Out* pois se baseia no comportamento para chegar ao modelo de processos, ou seja uma representação de fora para dentro. Frequentemente o *Play-In* é referido como inferência. Alguns exemplos de *Play-In* são os algoritmos de descoberta de processos como, α*-algorithm, Heuristics Miner, Inductive Miner*, etc.

- ***Replay:*** se fundamenta na reexecução do *event log* baseado no modelo de processo, e verifica a veracidade dos *logs* gerados quanto aos dados reais. Geralmente é utilizado na técnica de verificação *Conformance checking*.

A Tabela 2 apresenta um resumo dos conceitos de *Play-Out, Play-In* e *Replay*, demonstrando o conceito, o direcionamento de cada conceito, sua principal função, e por fim se o conceito requer *log* ou o modelo em seu funcionamento.

Conceito	Direção	Função Principal	Requer *Log*?	Requer Modelo?
Play-in	*Log* → Modelo	Descobrir o processo	Sim	Não
Play-out	Modelo → *Log*	Simular execuções possíveis	Não	Sim
Replay	*Log* ↔ Modelo	Comparar modelo com a execução real	Sim	Sim

Tabela 2 – Resumo dos conceitos de *Play-in, Play-out* e *Replay*.

Para armazenar os dados dos *event logs* o antigo formato padrão era o MXML *(Mining eXtensible Markup Language*[5]*)* e foi utilizado até o ano de 2010. Seu lançamento foi originado em 2003 e logo foi adotado como formato padrão da ferramenta de mineração de processos, ProM[6]. Com o uso desse formato é possível utilizar a sintaxe do XML *(Extensible Markup Language*[7]*)*. Porém, em Setembro de 2010 o MXML foi desbancado pelo XES *(eXtensible Event Stream*[8]*)*, após diversas experiências práticas com o formato MXML o padrão XES foi se tornando cada vez mais extensível e adaptativo em comparação ao MXML, se tornando mais atrativo, até que a *IEEE Task Force on Process Mining* adotou o novo formato de vez, para ser o formato padrão de troca para mineração de processos. Mais informações e detalhes sobre o formato XES podem ser encontradas em (IEEE Task Force on Process Mining, 2025).

[5] Linguagem de Marcação Extensível de Mineração
[6] ProM: ferramenta open-source para mineração e análise de processos de negócio. Disponível em: <https://promtools.org/>. Acesso em: 4 jun. 2025.
[7] Linguagem de Marcação Extensível
[8] Fluxo de Eventos Extensível

3 Trabalhos relacionados

Neste capítulo será apresentado o levantamento dos trabalhos que foram tomados como ponto de partida para este projeto de pesquisa. A busca teve como propósito encontrar trabalhos que trouxessem aspectos relacionados ao tema da pesquisa. Ela foi realizada em diferentes bases de dados digitais (*ACM Digital Library*, IEEE, Portal de periódicos CAPES, Springer e Google Acadêmico), incluindo palavras como "mineração de processos/*process mining*", "*WorkFlow net* interorganizacional/*interorganizational WorkFlow net*", "mineração de processos em processos de negócio/*process mining WorkFlow*", entre outras. Os trabalhos aqui apresentados contribuem com a justificativa desta pesquisa ao corroborar com o entendimento do campo de pesquisa tangente a mineração de processos em *WorkFlow nets* interorganizacionais.

3.1 Levantamento Bibliográfico

O trabalho de Noce (2017) [1] apresenta a modelagem de *WorkFlow nets* interorganizacionais usando a ferramenta CPN Tools. O objetivo do trabalho é demonstrar como duas redes distintas simulando duas organizações podem ser modeladas seguindo a premissa do fluxo de trabalho entre as duas empresas, tornando possível uma análise detalhada do fluxo entre elas. A metodologia utilizada para validar o resultado da pesquisa constituiu em uma análise qualitativa e outra quantitativa, comparando os resultados a métodos já testados por outros trabalhos da literatura. Em particular, o autor utilizou o algoritmo de verificação da propriedade *Soundness* (apresentado em (AALST, 1998)) e o método de replicação, disponível pelo CPN Tools. Noce (2017) conclui que obteve resultados satisfatórios quanto a confiabilidade e desempenho em termos do tempo de execução dos fluxos de trabalho. Enquanto que a propriedade *Soundness* garantiu a corretude do modelo.

Já em Medeiros e Günther (2005) [2], é apresentado uma maneira de extrair informações a partir de um modelo em rede de Petri, onde técnicas de geração de *logs* são implementadas. O artigo apresenta uma metodologia que adiciona funções em linguagem Standard ML, responsáveis por gerar arquivos com os dados relacionados aos *logs* da função geradora do processo em questão. Juntamente com as funções é destacado o emprego do framework ProM Imports para realizar a unificação dos arquivos de *logs* através do *plugin* de importação CPN Tools gerando um arquivo no formato MXML, conhecido como formato padrão no âmbito de mineração de processos entre as ferramentas disponíveis para desempenhar essa função. O arquivo MXML é aplicado em uma ferramenta de mineração de processos com alguns *plugins* ativos para retornar perspectivas da rede provenientes

dos *logs* fornecidos pelas funções do CPN Tools, validando todo o procedimento feito para aplicar mineração de processos em modelos de fluxo de trabalho através do CPN tools.

Por outro lado, Aalst (2011a) [3] aborda os conceitos relacionados as diferentes formas de mineração de processos, aplicando-as em duas abordagens distintas. O objetivo principal é demonstrar os problemas enfrentados ao lidar com ambientes interorganizacionais (onde se tem duas ou mais organizações envolvidas) e a aplicação de mineração de processos nesse modelo de trabalho. Partindo do pressuposto no qual a dificuldade é lidar com sistemas distribuídos, são apresentadas duas soluções: O particionamento horizontal e o particionamento vertical. O particionamento horizontal consiste na fragmentação dos processos entre as instituições, tornando cada uma responsável por um fragmento do procedimento. Já o particionamento vertical compartilha a execução dos casos de maneira semelhante, onde cada uma das instituições obtêm suas atribuições, sendo possível aplicar comparativos entre as entidades com o objetivo de testar a performance de cada execução. De acordo com as abordagens apresentadas em Aalst (2011a), é possível observar que a mineração de processos em *WorkFlows* interorganizacionais são benéficas ao permitir o compartilhamento de informações entre as empresas correlacionadas, bem como a comparação das informações que trafegam entre si.

Por sua vez, Engel et al. (2016) [4] abordam em seu trabalho a dificuldade de aplicar mineração de processos em modelos com dados baseados em EDI (Intercâmbio Eletrônico de Dados). Para isso, os autores identificam que podem ser obtidos diversos dados valiosos para as empresas. Porém, Engel et al. (2016) afirmam que até onde se sabe, não existe nenhuma abordagem que executa a extração desses dados. Sendo assim, os autores propuseram a criação de um *framework* chamado EDImine Framework. A proposta desta ferramenta é possibilitar a:

I - Aplicação de mineração de processos interorganizacionais suportados por EDI.

II - A avaliação de desempenho interorganizacional por meio de KPIs (Indicador-Chave de Desempenho) extraídos de mensagens EDI, *logs* de eventos e modelos de processo.

III - O pré-processamento semântico de mensagens EDIFACT para explorar essa fonte de informação.

Para a criação do *framework* foi proposto um modelo de pesquisa baseado em 3 perguntas que refletem o estado da arte do problema. Cada uma das perguntas forneceu o caminho necessário para alcançar o objetivo final. A ferramenta criada é composta por 3 componentes principais resultantes da pesquisa realizada. Ao final, Engel et al. (2016) criaram o conjunto de ferramentas EDImine Framework, e o *plugin* EDImine Framework BSC para realizar a análise dos dados obtidos na ferramenta de mineração de processos ProM 6. Quanto aos resultados, foram promissores, revelando informações valiosas do processo

interorganizacional em dados de uma empresa do mundo real. As limitações descobertas estão ligadas ao fato de algumas informações essenciais do fluxo natural de uma empresa não serem compostos por mensagens EDI, portanto alguns dos processos não podem ser analisados.

Por fim, a pesquisa de Aouachria et al. (2024) [5] é voltada para o estudo de incompatibilidades na *WorkFlow net* interorganizacional. O objetivo do trabalho é projetar e desenvolver um método de identificação de incompatibilidades. Após identificá-las, os autores realizam a adaptação de processos para resolver as incompatibilidades entre os processos de parceiros. A metodologia usada na criação do modelo de resolução de incompatibilidades denominado PM4BPI *(Process Mining for Business Process Integration)*, utiliza um algoritmo de árvore de decisão proposto em (AALST, 2011b). O método proposto consegue realizar a identificação de 6 tipos diferentes de incompatibilidades de processos. Assim que o algoritmo identifica a incompatibilidade, aplica o modelo de adaptação de processo necessário para o tratamento da incompatibilidade. Para validar os resultados do método, o trabalho foi aplicado em um exemplo do mundo real usando uma unidade de atendimento da Argélia que envolve duas clinicas de saúde. Quanto aos resultados, Aouachria et al. (2024) afirmam que foram promissores, resolvendo a incompatibilidade e aplicando a adaptação de processos. Porém, os autores também afirmam que ainda tem de ser feitos mais testes, e que somente são tratadas 6 das incompatibilidades, mesmo que sejam as mais comuns e frequentes, ainda é cedo para afirmar com precisão sua eficácia. Outra observação feita por Aouachria et al. (2024) é sobre o nível de complexidade que os projetistas tem que lidar para aplicar este método em processos de negócios incompatíveis, e que o nível de experiência elevado é essencial.

3.2 Considerações Finais

Em resumo, os trabalhos relacionados tem diversos aspectos em comum, mas ao analisar a Tabela 3 observa-se que nenhum dos trabalhos abordam todos os requisitos necessários para uma metodologia que consiste em simular, analisar, e aplicar mineração de processos em *WorkFlow nets* interorganizacionais. Para a construção dessa metodologia, os trabalhos relacionados contribuíram com aspectos importantes. Por exemplo, o trabalho de Noce (2017) (Trabalho 1, na Tabela 3) estabelece a formalização da *WorkFlow net* interorganizacional na ferramenta CPN Tools, apresentando assim uma forma de usar a ferramenta tanto para edição quanto para simulação da rede. Já a pesquisa de Medeiros e Günther (2005) (Trabalho 2) apresenta como aplicar as funções que serão responsáveis por gerar os *logs* de eventos extraídos de redes simuladas no CPN Tools. Após a junção destes em único arquivo, é possível usar ferramentas de mineração de processos para analisar o processo modelado. Com a contribuição dos trabalhos de Aalst (2011a) (Trabalho 3), Engel

	Trabalho [1]	Trabalho [2]	Trabalho [3]	Trabalho [4]	Trabalho [5]	Este trabalho
Mineração de Processos	✘	✓	✓	✓	✓	✓
Modelagem	✓	✓	✘	✘	✓	✓
Extração de Arquivos de *Logs* com a Junção em Formato Padrão Para Mineração de Processos	✘	✓	✘	✓	✓	✓
Rede de Petri colorida	✓	✓	✘	✘	✘	✓
Simulação Usando CPN Tools	✓	✓	✘	✘	✘	✓
Workflow net Interorganizacional	✓	✘	✓	✓	✓	✓

Tabela 3 – Comparação dos trabalhos relacionados.

et al. (2016) (Trabalho 4) e Aouachria et al. (2024) (Trabalho 5), é possível a identificação de problemas ligados ao fluxo de processos interorganizacionais. Em particular, esses trabalhos apresentam estudos em organizações reais. Com isso torna-se possível estabelecer critérios para a análise dos modelos, com base em problemas recorrentes nesse tipo de processo.

Neste contexto, a presente pesquisa contribui para a área de estudo ao apresentar uma alternativa que permite às organizações especificar seus processos, simular diferentes cenários e analisá-los com o objetivo de identificar inconsistências e possíveis problemas.

4 Modelo WorkFlow net Interorganizacional

4.1 Escolha do Modelo

O primeiro passo para a execução da proposta deste trabalho consiste na realização do objetivo específico responsável por definir um processo interorganizacional, representando as atividades, as entidades colaboradoras e seus fluxos de interação. Portanto, a escolha do modelo foi baseada no trabalho de Aalst (1999). Dessa forma o modelo escolhido para ser utilizado neste trabalho foi a rede apresentada na Figura 12.

Em Aalst (1999), o modelo não foi associado a nenhuma narrativa específica. Sendo assim, ao ser implementado neste trabalho, foi associado ao modelo uma narrativa para fins de testes. Essa narrativa permitiu gerar informações a respeito do processo e, a partir disso, extrair os dados para análise por meio da mineração de processos.

4.2 Modelo IOWF-*net* de Simulação

A narrativa criada para o processo da Figura 12, foi baseada na descrição feita por Aalst (1999) onde apresenta que o modelo segue o particionamento horizontal com a abordagem fracamente acoplada representando a comunicação entre produtor e fornecedor. O modelo foi então adaptado para uma rede de Petri Colorida através do CPN Tools. Para este fim, o modelo foi dividido em duas redes locais chamadas LWF1 e LWF2. A rede LWF1 representa o fluxo de trabalho do comprador, e a rede LWF2 do fornecedor, como pode ser visto na Figura 16.

A alteração do nome das transições do modelo foi a primeira a ser realizada. O nome ajudará a identificar a atividade e as informações associadas. Portanto, agora as transições que na Figura 12 seguiam a nomenclatura de T1, T2, T3, etc., foram alteradas para representar as atividades que estão sendo realizadas no modelo, por exemplo, Criar Requisicao, Reavaliar Requisicao, Aprovar Requisicao, etc., como ilustra a Figura 16. A ferramenta CPN Tools não tem suporte a acentuação, por este motivo os nomes não estão acentuados devidamente.

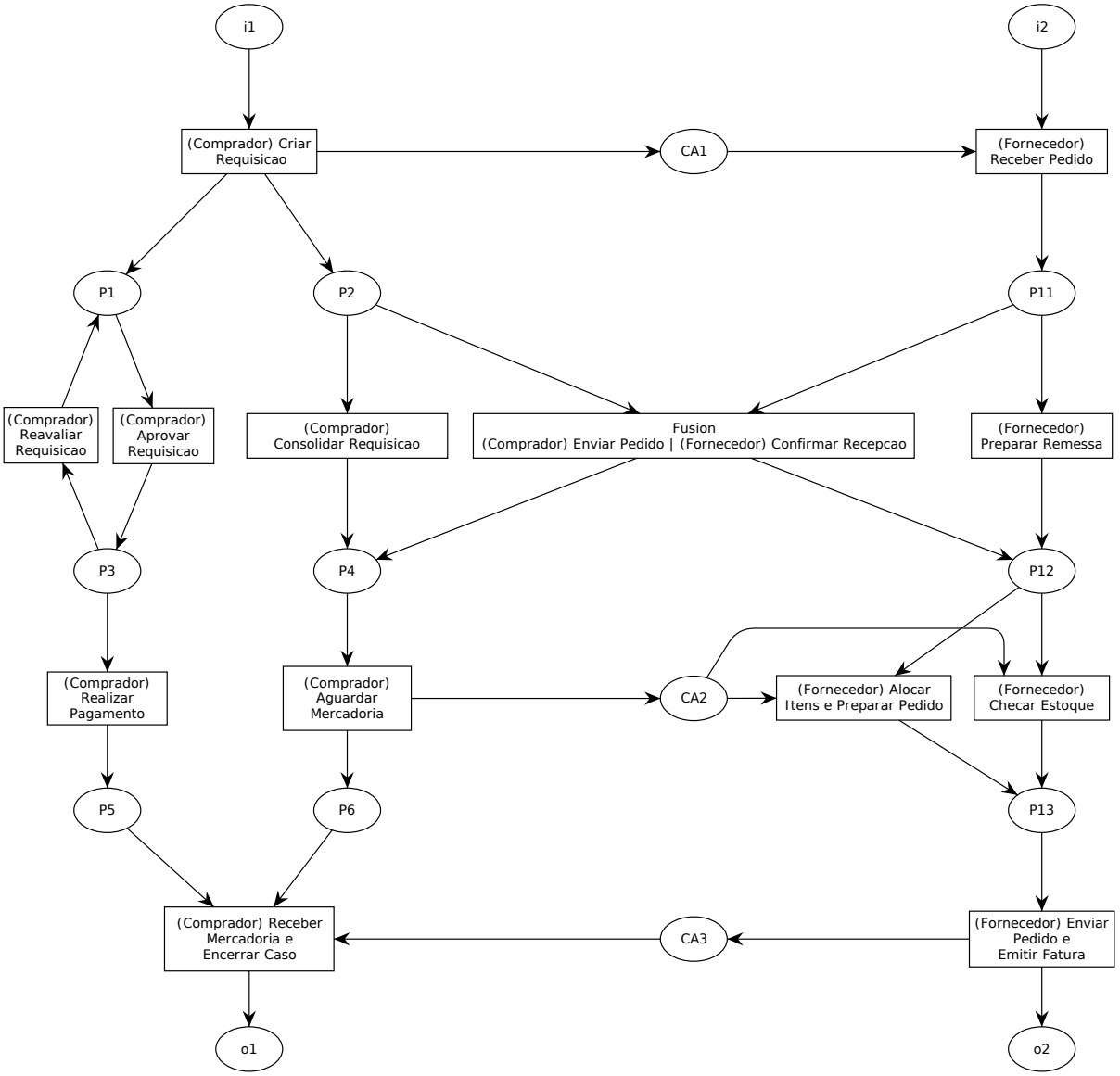


Figura 16 – IOWF-*net* adaptada de Aalst (1999).

A Tabela 4 representa um resumo da troca do antigo nome das transições (Figura 12) para o novo nome atribuído. Observa-se também que para cada atividade existe um identificador entre parenteses antes do nome da transição, demonstrando de qual rede pertence cada uma das atividades. As transições T5 e T12, na Tabela 4, estão com uma observação na frente do nome. Isso acontece porque essas duas transições representam uma comunicação síncrona entre a LWF1 e LWF2, ou seja, elas precisam disparar ao mesmo tempo. Por isso, no modelo da Figura 16 essas duas transições são representadas como uma só. Apesar disso, o disparo da transição gerará informações sobre as duas atividades correspondentes.

Nome Antigo	Nome Novo (Atividade)
T1	(Comprador) Criar Requisicao
T2	(Comprador) Reavaliar Requisicao
T3	(Comprador) Aprovar Requisicao
T4	(Comprador) Consolidar Requisicao
T5	(Comprador) Enviar Pedido (Fusão com T12)
T6	(Comprador) Realizar Pagamento
T7	(Comprador) Aguardar Mercadoria
T8	(Comprador) Receber Mercadoria e Encerrar Caso
T11	(Fornecedor) Receber Pedido
T12	(Fornecedor) Confirmar Recepcao (Fusão com T5)
T13	(Fornecedor) Preparar Remessa
T14	(Fornecedor) Checar Estoque
T15	(Fornecedor) Enviar Pedido e Emitir Fatura
T16	(Fornecedor) Alocar Itens e Preparar Pedido

Tabela 4 – Resumo da troca de nomes das transições do modelo da Figura 12 para a Figura 16

Agora que o nome das atividades já foi definido e alterado devidamente, deve ser alterado o formato da ficha inicial que irá percorrer o modelo. Para isso devem ser declarados tipos de dados (*colsets*) e variáveis na ferramenta CPN Tools.

A ficha de marcação inicial é o elemento que representa o caso sendo executado através do modelo, portanto, ela deve carregar todos os elementos que o caso em específico registra ou altera ao longo da execução do processo. Como o modelo está sendo adaptado para que seja possível realizar a mineração de processos, a ficha inicial para se tornar mais dinâmica, foi transformada em uma função na qual recebe apenas um parâmetro. Este parâmetro é um número N que representa a quantidade de fichas que o modelo vai receber. Na Figura 17 no lugar inicial “i1”, da rede LWF1, pode ser observado que a ficha inicial é uma função chamada “criar_tokensC 10” e no lugar “i2”, que representa o início da LWF2, a ficha inicial é “criar_tokensF 10”. Ambas resultam na criação de 10 fichas para cada rede local. No entanto há diferença nas fichas geradas, pois algumas informações são específicas de cada modelo, por este motivo devem ser criadas duas funções diferentes. Para que o modelo funcione de maneira correta deve ser inserida a mesma quantidade de fichas em ambas as redes.

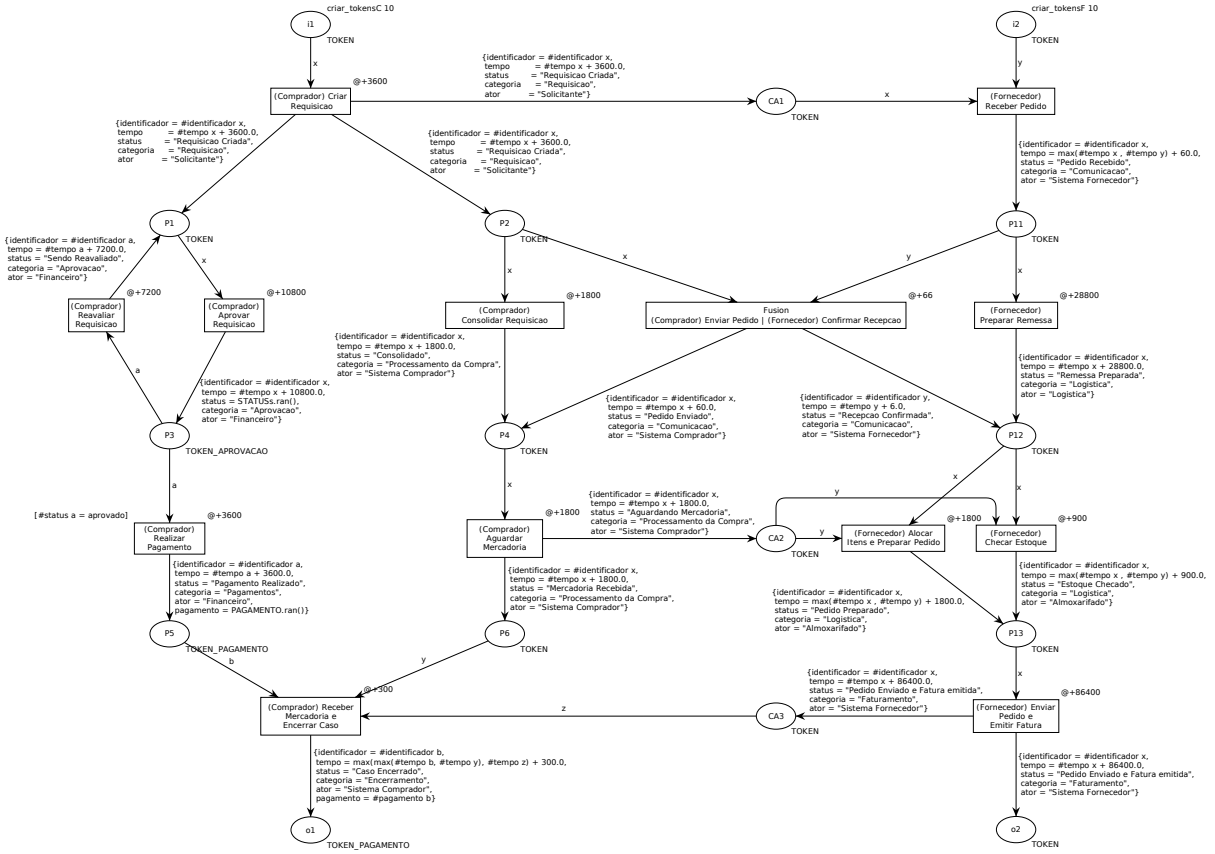


Figura 17 – IOWF-*net* com os tipos de dados associados representando as informações do processo.

As funções que geram as fichas são compostas por uma expressão CPN ML da ferramenta CPN Tools que cria uma lista tabulada de `N` elementos, conforme apresentado a seguir:

```
List.tabulate(tamanho,função_geradora)
```

O valor "`tamanho`" é a quantidade de elementos, e a `função_geradora` para a LWF1 é dada pela seguinte expressão:

```
(fn i => {identificador = i + 1, tempo = Time.toReal(Time.now()),
status = "Criado", categoria = "Requisicao", ator = "Solicitante"})
```

Já a função `função_geradora` para a rede LWF2 é dada pela seguinte expressão:

```
(fn i => {identificador = i + 1, tempo = Time.toReal(
Time.now()), status = "Fornecedor Pronto", categoria
= "Inicializador", ator = "Sistema Fornecedor"})
```

Para uma lista de tamanho `N`, o índice `i` começa em 0, é incrementado em 1 a cada elemento até o último valor de `i` ser `N-1`. Dessa forma, quando o usuário determinar uma quantidade maior ou menor de fichas para ser gerada, será necessário apenas alterar o parâmetro de cada função.

Para gerar este tipo de ficha, o lugar inicial juntamente com todos os outros lugares por onde essa ficha percorrer devem pertencer ao tipo de dado correspondente a ficha. Por padrão ao criar um lugar na rede dentro do CPN Tools, ele já vem com o *color set* definido como *Unit*. Porém, as fichas do modelo precisam de um tipo mais complexo do que esse para carregar todas as informações do processo. Na próxima seção serão apresentadas todas as declarações de tipos, funções e variáveis implementadas para o modelo da Figura 17.

4.2.1 Funções CPN ML do Modelo de Simulação

As funções abaixo escritas em CPN ML, são utilizadas para a manipulação de dados, conversão de tipos e inicialização da rede com listas de fichas.

- `fun max(a:real, b:real) = if a > b then a else b;`
  - **Nome:** `max`.
  - **Parâmetros:** `a` e `b` (ambos do tipo `real`).
  - **Retorno:** tipo `real`.
  - **Ação:** compara dois números e retorna o maior valor entre eles; utilizado no modelo, pois o tempo considerado pelo CPN Tools é sempre o de maior valor. Em alguns trechos do modelo, como representado na Figura 17, mais de uma atividade é executada, e portanto dois tempos de execução de transição diferentes são recebidos, porém apenas o maior entre eles deve ser armazenado. Este valor é armazenado em uma variável.

- `fun statusToString(aprovado) = "aprovado" | statusToString(reprovado) = "reprovado";`
  - **Nome:** `statusToString`.
  - **Entrada:** um valor do conjunto enumerado `STATUSs`.
  - **Ação:** realiza uma conversão de tipos, convertendo o status enumerado em sua representação textual (`string`). Pois quando as funções geradoras de *logs* forem inseridas no modelo, este valor deverá ser inserido em um campo reservado para `string`.
  - **Mapeamento:** `aprovado` → “aprovado” | `reprovado` → “reprovado”.

- `fun pagamentoToString(realizado) = "realizado" | pagamentoToString (atrasado) = "atrasado";`
  - **Nome:** `pagamentoToString`.
  - **Entrada:** um valor do conjunto enumerado `PAGAMENTO`.
  - **Ação:** converte o status do pagamento enumerado para `string`. Da mesma forma que a função anterior.
  - **Mapeamento:** `realizado` → "realizado" | `atrasado` → "atrasado".

- `fun criar_tokensC(n) = List.tabulate(n,fn i => {identificador = i + 1, tempo = Time.toReal(Time.now()), status = "Criado", categoria = "Requisicao", ator = "Solicitante"});`
  - **Nome:** `criar_tokensC` (Fichas Comprador).
  - **Parâmetro:** `n` (inteiro representando a quantidade de fichas).
  - **Ação:** utiliza a função `List.tabulate` para gerar uma lista de fichas iniciais para o fluxo do comprador.
  - **Definições Fixas:** define `status="Criado"`, `categoria="Requisicao"` e `ator="Solicitante"`. O identificador é gerado sequencialmente (`i+1` até `n`).

- `fun criar_tokensF(n) = List.tabulate(n, fn i => {identificador = i + 1, tempo = Time.toReal(Time.now()), status = "Fornecedor Pronto" , categoria = "Inicializador", ator = "Sistema Fornecedor"})`
  - **Nome:** `criar_tokensF` (Fichas Fornecedor).
  - **Parâmetro:** `n` (inteiro representando a quantidade de fichas).
  - **Ação:** semelhante à função anterior, mas gera fichas para o fluxo do Fornecedor.
  - **Definições Fixas:** define `status="Fornecedor Pronto"`, `categoria="Inicializador"` e `ator="Sistema Fornecedor"`. O identificador também é gerado sequencialmente (`i+1` até `n`).

4.2.2 Declarações dos color sets

Os conjuntos de cores definidos para o modelo estabelecem os tipos de dados básicos e enumerados (*enumerated types*) que serão utilizados.

- `colset ID = int;`

- **Nome:** `ID`.
- **Definição:** um conjunto de cores que é um alias (apelido) para o tipo inteiro (`int`).
- **Uso:** será usado para campos de identificação única nas fichas.

- `colset TEMPO = real;`
  - **Nome:** `TEMPO`.
  - **Definição:** um alias para o tipo real (`real`).
  - **Uso:** usado para armazenar o tempo de simulação.

- `colset STATUS = string;`
  - **Nome:** `STATUS`.
  - **Definição:** um alias para o tipo string (`string`).
  - **Uso:** usado para o *status* geral de uma ficha (ex.: "Criado", "Em Andamento").

- `colset STATUSs = with aprovado | reprovado;`
  - **Nome:** `STATUSs`.
  - **Definição:** um conjunto de cores de tipo enumerado, onde os únicos valores possíveis são `aprovado` e `reprovado`. Este é o tipo usado pela função `statusToString`.
  - **Uso:** usado nas fichas que passaram por um processo de aprovação para gerar um resultando randômico.

- `colset CATEGORIA = string;`
  - **Nome:** `CATEGORIA`.
  - **Definição:** um alias para o tipo *string* (`string`).
  - **Uso:** usado para classificar a categoria da ficha em determinados pontos do modelo (ex.: "Requisicao", "Inicializador").

- `colset ATOR = string;`
  - **Nome:** `ATOR`.
  - **Definição:** um alias para o tipo *string* (`string`).
  - **Uso:** usado para identificar a entidade que criou ou está processando a ficha (ex.: "Solicitante", "Sistema Fornecedor").

- `colset PAGAMENTO = with realizado | atrasado;`
  - **Nome:** `PAGAMENTO`.
  - **Definição:** um conjunto do tipo enumerado, com os valores possíveis `realizado` e `atrasado`. Este é o tipo usado pela função `pagamentoToString`.
  - **Uso:** usado nas fichas que passaram por um processo de pagamento para gerar um resultando randômico.

4.2.3 Declarações das fichas

Nesta seção são definidos os *color sets* compostos, utilizando o tipo `record` que é uma espécie de registro/estrutura.

- ```
  colset TOKEN = record
    identificador : ID *
    tempo : TEMPO *
    status : STATUS*
    categoria : CATEGORIA *
    ator : ATOR
  timed;
  ```
  - **Nome:** `TOKEN`.
  - **Definição:** define o formato principal de uma ficha. Ele contém campos com os tipos declarados na subseção 4.2.2:
    - `identificador : ID *`
    - `tempo : TEMPO *`
    - `status : STATUS *`
    - `categoria : CATEGORIA *`
    - `ator : ATOR *`
  - **Detalhe:** o asterisco (`*`) após o tipo é uma sintaxe do CPN ML que indica que o tipo é não-ordenado, utilizado para registros.
  - **Timed:** `timed;`, indica que as fichas desse *color set* são sensíveis ao tempo na simulação.

- ```
  colset TOKEN_APROVACAO = record
    identificador : ID *
  ```

```
  tempo : TEMPO *
  status : STATUSs *
  categoria : CATEGORIA *
  ator : ATOR
timed;
```

- **Nome:** `TOKEN_APROVACAO`.
- **Definição:** uma estrutura para fichas semelhante ao `TOKEN`, mas com o campo `status` usando o conjunto de cores `STATUSs` (`aprovado | reprovado`):
  - ▷ `status : STATUSs *`
- **Uso:** representa uma ficha que está no estágio de aprovação nas transições do modelo.

- `colset TOKEN_PAGAMENTO = record`
  `identificador : ID *`
  `tempo : TEMPO *`
  `status : STATUS *`
  `categoria : CATEGORIA *`
  `ator : ATOR *`
  `pagamento : PAGAMENTO`
  `timed;`
  - **Nome:** `TOKEN_PAGAMENTO`.
  - **Definição:** semelhante aos *color sets* anteriores, porém adiciona um campo específico para rastrear o pagamento:
    - ▷ Inclui todos os campos do `TOKEN`.
    - ▷ Adiciona: `pagamento : PAGAMENTO *`
  - **Uso:** representa uma ficha que está no estágio de pagamento nas transições do modelo.

4.2.4 Declarações de Variáveis

As variáveis são usadas nas inscrições dos arcos e nas guardas das transições para manipular as fichas e verificar condições para disparo das transições.

- `var id : ID;`

- **Nome:** `id`.
- **Tipo:** `ID` (inteiro).
- **Uso:** usada para ligar o identificador de uma ficha ou para gerar um novo `ID`.

- `var x, y, z : TOKEN;`
  - **Nomes:** `x`, `y`, `z`.
  - **Tipo:** `TOKEN` (o registro principal criado anteriormente).
  - **Uso:** usadas para ligar as fichas do tipo `TOKEN` entre os lugares da rede.

- `var a : TOKEN_APROVACAO;`
  - **Nome:** `a`.
  - **Tipo:** `TOKEN_APROVACAO`.
  - **Uso:** usada para ligar as fichas do tipo `TOKEN_APROVACAO`.

- `var b : TOKEN_PAGAMENTO;`
  - **Nome:** `b`.
  - **Tipo:** `TOKEN_PAGAMENTO`.
  - **Uso:** usada para ligar as fichas do tipo `TOKEN_PAGAMENTO`.

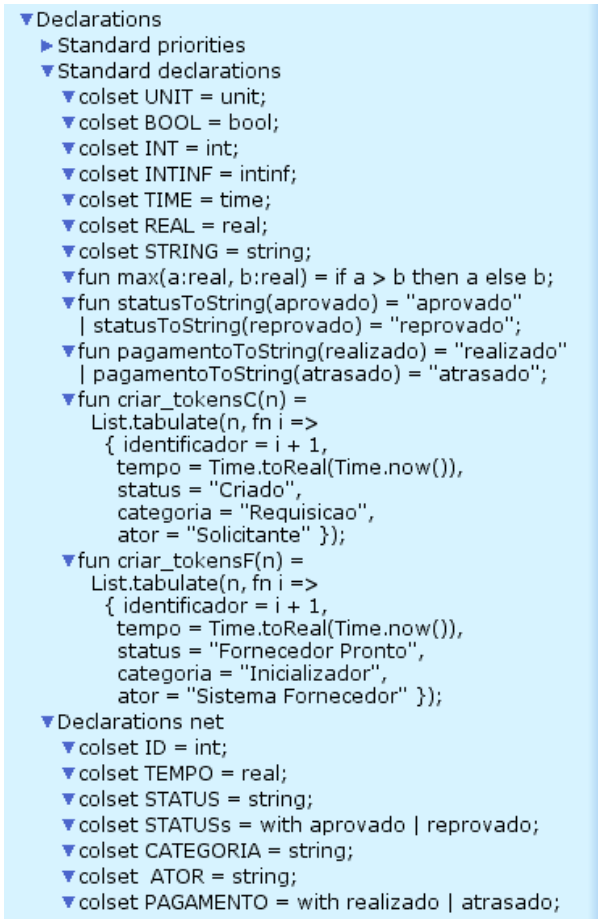
4.2.5 Declarações de Logs

As declarações de *logs* definem constantes de valor (*`val`*) e importações de código (*`use`*), tipicamente relacionadas à funcionalidade de *logs* da simulação. Este trecho está diretamente ligado a próxima seção do trabalho, onde será demonstrado a adaptação do modelo para receber as funções geradoras de *logs* de eventos.

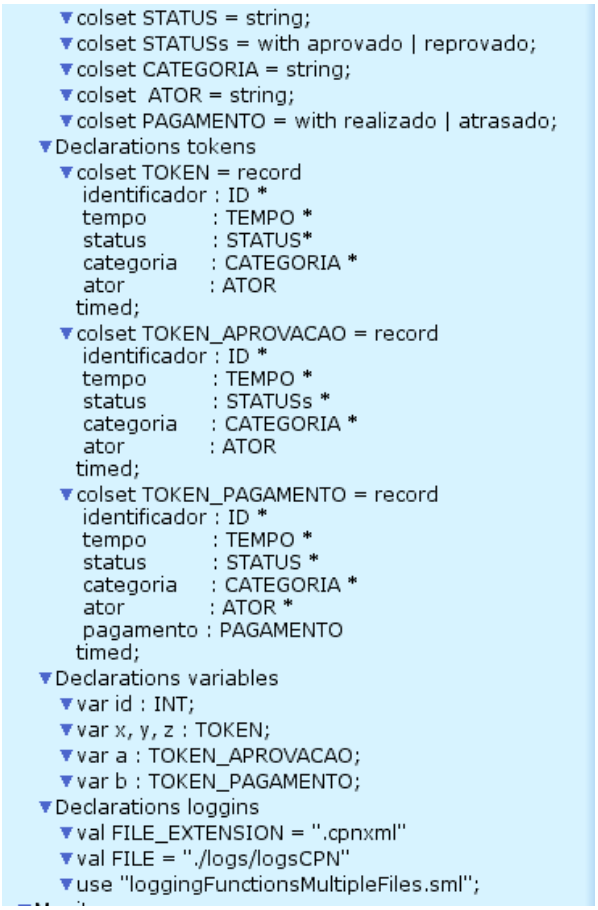
- `val FILE_EXTENSION = ".cpnxml";`
  - **Tipo:** declaração de valor constante (`val`).
  - **Nome:** `FILE_EXTENSION`.
  - **Valor:** a string `".cpnxml"`.
  - **Uso:** define a extensão de arquivo padrão para cada um dos arquivos de *logs* de evento que serão gerados por cada caso quando o modelo executar a simulação.

- `val FILE = "./logs/logsCPN";`
  - **Tipo:** declaração de valor constante (`val`).
  - **Nome:** `FILE`.
  - **Valor:** a string `"./logs/logsCPN"`.
  - **Uso:** define o caminho base para armazenamento dos arquivos e o nome para o arquivo de *log* de evento.

- `use "loggingFunctionsMultipleFiles.sml";`
  - **Tipo:** diretiva de módulo (`use`).
  - **Ação:** carrega e executa o código contido no arquivo externo.
  - **Uso:** importa funções customizadas para geração de *logs* em múltiplos arquivos.



(a) Primeira parte.



(b) Segunda parte.

Figura 18 – Seção de declarações da ferramenta CPN Tools.

Com todas as configurações realizadas, o modelo na ferramenta CPN Tools irá gerar a ficha, como ilustrado pela Figura 19. Nela é possível observar que a ficha agora é criada

através da função que foi demonstrada na subseção 4.2.1. Pode ser observado que o lugar inicial "i1" é do tipo `TOKEN` e o arco que liga este lugar a transição "(Comprador) Criar Requisicao" tem um único elemento, que é a variável `x`, a qual suporta a ficha criada, pois tem o mesmo *color set*. Na transição destaca-se o elemento `@+3600` que representa o tempo em segundos que essa transição demora para ser disparada. Este tempo que é armazenado de acordo com a função `fun max(a:real, b:real) = if a > b then a else b;`, quando existem duas ou mais transições com arcos de saída que tem tempos diferentes. Já no arco de saída da transição "(Comprador) Criar Requisicao" para o lugar "P1", observa-se que alguns valores estão sendo atualizados, como o tempo e o status, já quando os valores se mantêm o campo recebe ele mesmo como valor. Para fazer isso, quando se necessita acessar um campo específico de uma estrutura `record`, usa-se o símbolo "#" seguido do nome do campo e a variável que o armazena, nesse caso `x`, dessa forma o trecho fica da seguinte forma: `identificador = #identificador x,`, como demonstrado na Figura 19.

Outro elemento importante que pode ser destacado na Figura 19, é o tempo, que recebe um valor do tipo `real` através da função `Time.toReal(Time.now())` com um número que representa a data atual toda vez que a ferramenta CPN Tools é aberta para que o modelo armazene a data atual na simulação. Caso o usuário necessite que o valor seja representado no formato de data, basta apenas que utilize uma função de manipulação de datas como preferir.

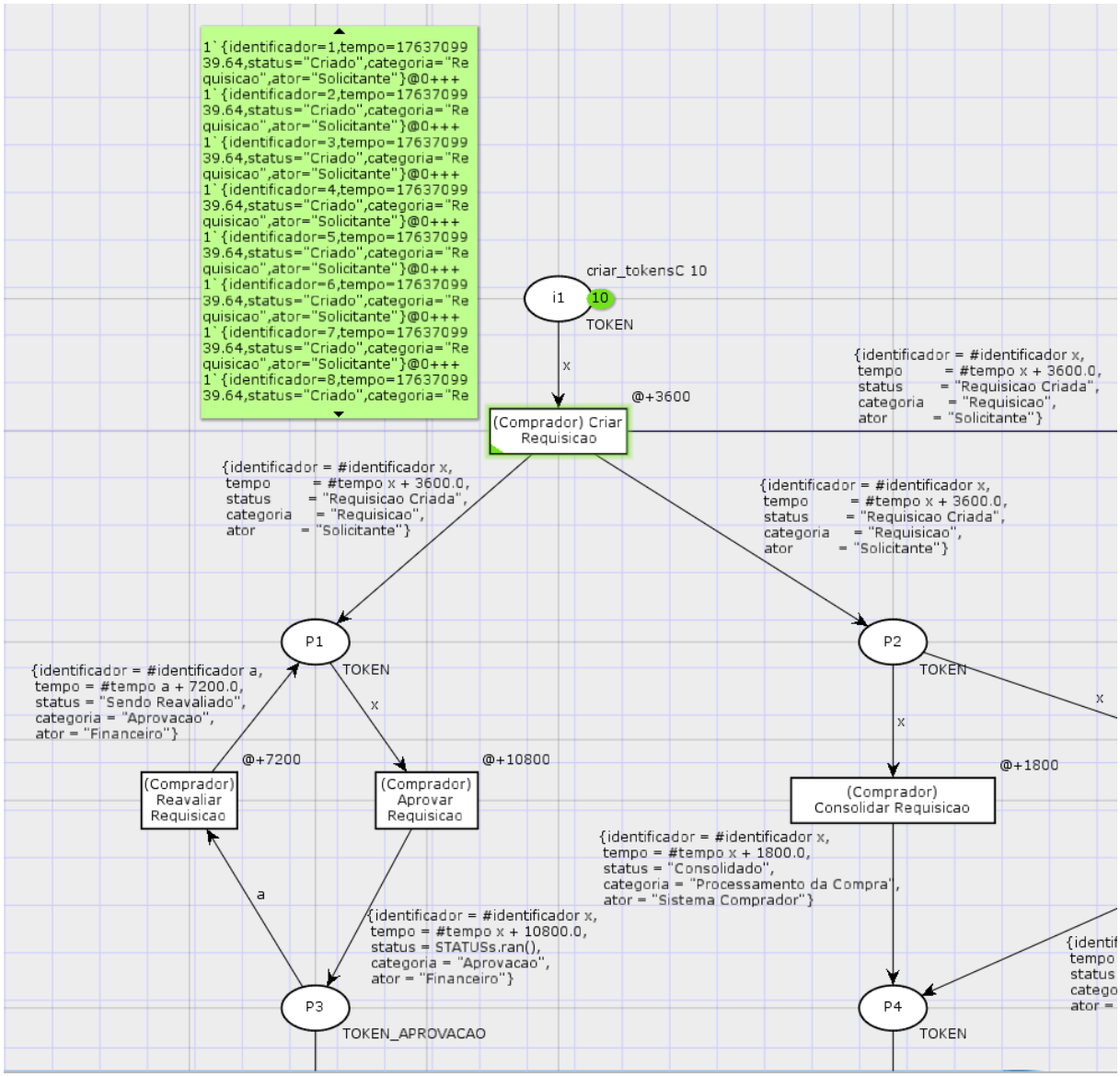


Figura 19 – Estado inicial da rede criada no CPN Tools.

4.3 Inserção das Funções Geradoras de *Logs*

Uma vez que o modelo foi definido e as informações estabelecidas, agora é preciso realizar a adaptação do modelo, para que seja possível inserir as funções geradoras de *logs*. O registro dos logs a partir da rede será feito conforme apresentado por Medeiros e Günther (2005). O primeiro passo é o passo demonstrado na subseção 4.2.5, conforme representado na Figura 18b. Após inserir esses dados, e criar o diretório *logs* no caminho raiz, onde o arquivo do modelo está localizado, o arquivo "loggingFunctionsMultipleFiles.sml" deve ser adicionado no mesmo diretório da rede implementada no CPN Tools.

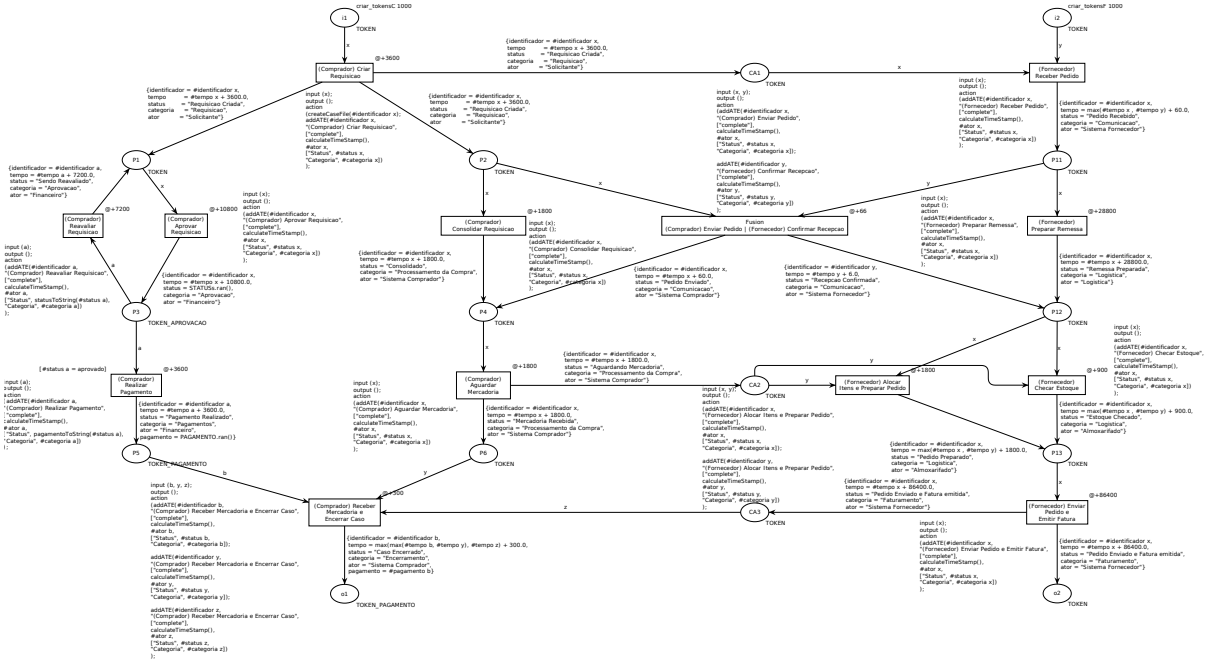


Figura 20 – IOWF-*net* com funções geradoras de *logs* inseridas.

4.3.1 Configuração de Log e Transições

Após a atualização das declarações da rede, as inscrições de entrada, saída e ação das transições devem ser modificadas para permitir chamar as funções de registro (*logging*). O objetivo é que a rede crie um *log* MXML parcial para cada caso executado. Para isso, são disponibilizadas duas funções principais em ML: `createCaseFile` e `addATE`.

4.3.2 Funções de Registro (Logging Functions)

- `fun createCaseFile(caseId)`
  - **Descrição:** abre o arquivo de *log* para um caso específico.
  - **Regra:** deve ser chamada apenas uma vez por caso e obrigatoriamente antes da função `addATE` ser chamada para este mesmo caso.
  - **Entrada:** recebe um número inteiro (`int`) que representa o identificador único do caso.
  - **Exemplo:** normalmente utilizada em uma transição geradora (como *Generator*) para iniciar o rastreamento. Porém, como neste trabalho a geração das fichas ocorre por meio de uma função geradora, pode ser inserida na primeira transição do modelo, neste caso a transição "(Comprador) Criar Requisicao".

- `fun addATE(caseId, transitionName, eventType, timestamp, originator, data)`
  - **Descrição:** registra a execução de uma transição no *log* do caso correspondente.
  - **Parâmetros Detalhados:**
    1. **caseId:** inteiro (`int`) que identifica unicamente o caso.
    2. **transitionName:** string (`string`) contendo o nome da transição a ser registrada (ex: `"(Comprador) Criar Requisicao"`).
    3. **eventType:** lista de strings.
       - ▷ Se o tipo é conhecido, a lista contém um elemento: `["complete"]`, `["start"]`, `["suspend"]`, `["resume"]`, `["autoskip"]`, `["manualskip"]`, `["pi_abort"]`, e `["ate_abort"]`.
       - ▷ Se o tipo de evento é desconhecido, usa-se o formato `["unknown", "nome"]` onde nome é o tipo de evento desconhecido.
    4. **timestamp:** string no formato XML *dateTime* (ex: `"2005-06-30T14:55:00.000+01:00"`).
       - ▷ A função auxiliar `calculateTimeStamp()` disponível no arquivo *loggingFunctionsMultipleFiles.sml* pode ser usada para calcular automaticamente este campo baseando-se no tempo atual da simulação.
    5. **originator:** string com o nome do originador que pode ser uma pessoa ou o sistema que executou a tarefa (ex: `"Solicitante"`). Se vazio, passar `""` como argumento.
    6. **data:** lista de strings com dados adicionais associados à tarefa.
       - ▷ Formato: `[nomeAtributo1, valor1, nomeAtributo2, valor2, ..., nomeAtributoX, valorX]`.
       - ▷ Se não houver dados extras, deve-se passar uma lista vazia `[]`.
       - ▷ Na adaptação do modelo uma lista com dois elementos é passada como argumento das funções, no primeiro registro da função `addATE` pode-se observar na transição "(Comprador) Criar Requisicao" que existem duas informações extras `["Status", #status x, "Categoria", #categoria x]`.

A simulação do modelo com as novas funções geradoras criará arquivos de *logs* MXML parciais. Na Figura 20, ao observar as transições que antecedem ou procedem dos lugares que tem o *color set* `"TOKEN_APROVACAO"` ou `"TOKEN_PAGAMENTO"`, é possível identificar as funções `statusToString` e `pagamentoToString`, pois o valor que será armazenado no campo `#status` e `#categoria` deve ser uma `string`.

Outro detalhe presente na implementação das funções geradoras de *logs* é a implementação das funções `addATE` múltiplas, pois existem transições que são responsáveis por executar mais de uma função por vez, como é o caso da transição resultante da fusão entre as transições "(Comprador) Enviar Pedido" e "(Fornecedor) Confirmar Recepcao". Como ela precisa gerar dois registros independentes no *log* do `caseId`, é necessário apenas passar como parâmetro na função de entrada `input ();` os dois parâmetros que são `x` e `y`, uma para cada arco de entrada na transição, dessa forma é possível criar uma função `addATE` para cada uma das variáveis. Isso acontece em outros pontos do modelo quando é necessário criar um registro para cada variável distinta proveniente dos arcos de entrada que a transição recebe.

Na Figura 20 pode ser observado na transição "(Comprador) Realizar Pagamento" a existência de uma guarda. Guardas são usados para testes em variáveis de inscrição de arco de entrada habilitando restrições no disparo da transição. Neste caso a transição em questão só será disparada se a variável `#status a` for igual a `aprovado`, que é o valor enumerado do tipo `STATUSs`.

Após concluir a inclusão das funções em todas as transições do modelo conforme demonstrado na Figura 20, o próximo passo a ser executado para que seja possível realizar a mineração de processos é realizar a unificação dos *logs* MXML parciais em um único arquivo padrão de mineração. O próximo capítulo deste trabalho, demonstra como realizar essa etapa.

5 Análise do Modelo

5.1 Mineração de Processos

Após as etapas anteriores, já com os *logs* de eventos gerados, a tarefa a ser realizada para dar início a mineração de processos é proceder com a junção dos arquivos gerados anteriormente, produzindo um arquivo único. O formato desse arquivo foi definido pela plataforma *ProM Imports Framework*, por sua vez explicado detalhadamente na subseção 5.1.1. Para prosseguir com a mineração é necessário o auxílio de *plugins* que ao ser executado em conjunto com o arquivo gerado pelo *ProM Imports Framework*, gera uma série de resultados que revelam o comportamento do modelo IOWF-*net*, processo esse explicado detalhadamente na subseção 5.1.2.

5.1.1 ProM Imports Framework

O processo de unificação dos *logs* de eventos consiste na utilização da ferramenta *ProM Imports Framework* na versão 7.0. Ao abrir o programa a interface exibida pode ser vista na Figura 21. Na interface do programa ao localizar a aba *"Choose filter"* deve ser selecionado o filtro "CPN Tools", após selecioná-lo é preciso apontar o tipo de saída gerado pela ferramenta. Neste trabalho foi selecionado a opção "MXML", na *combo box "Output as"*, em seguida foi selecionado a pasta onde este arquivo será salvo, para escolher a pasta basta utilizar o botão *"change"*. Após realizar essas configurações iniciais, a próxima etapa consiste na configuração das propriedades do filtro selecionado, para isso seleciona-se a aba *"Filter Properties"*, conforme ilustra a Figura 22.

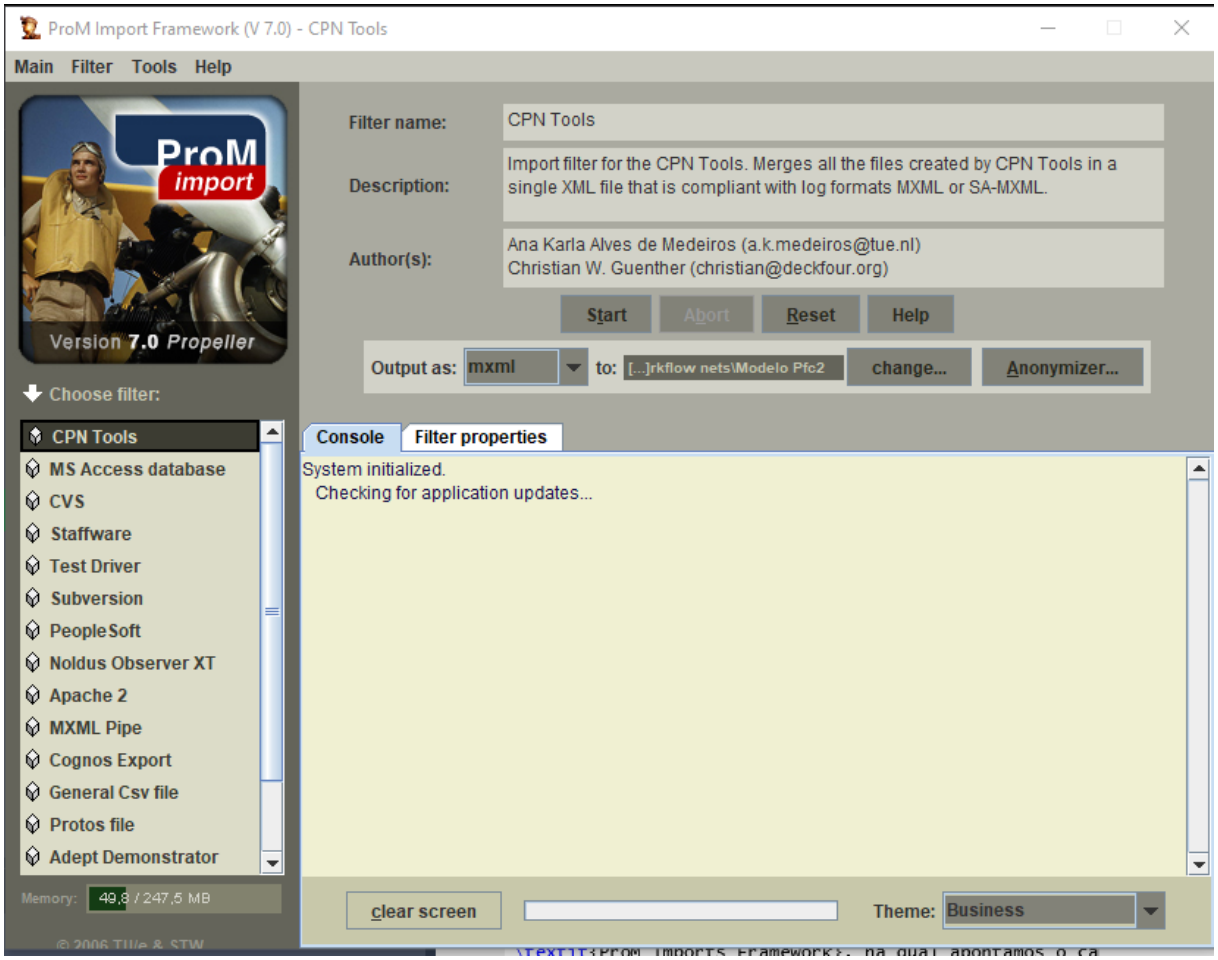


Figura 21 – *Screenshot* da interface do *ProM Imports Framework*.

Nesta etapa é preciso configurar a ferramenta da seguinte forma: para a propriedade *"CPN log files directory"*, o diretório onde estão localizados os *logs* de eventos gerados pelas funções que foram adaptadas ao modelo dentro da ferramenta CPN Tools deve ser selecionado; para a propriedade *"CPN log files suffix"* o formato "cpnxml" deve ser selecionado.

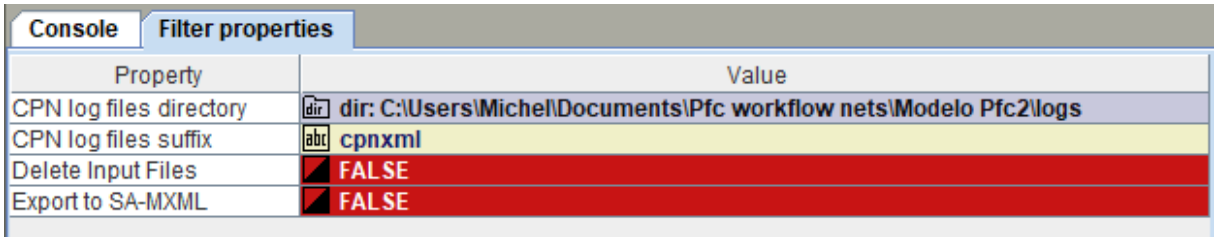


Figura 22 – *Screenshot* da aba de propriedades do filtro.

Com essas configurações, a ferramenta está pronta para iniciar a unificação dos *logs* de eventos em um arquivo no formato MXML. Assim, o botão "Start" deve ser acionado, com isso os passos gerados pela ferramenta devem ser atualizados na aba "Console". Por fim o arquivo gerado pode ser adicionado na pasta de destino selecionada.

5.1.2 ProM

Para seguir adiante com a mineração de processos, outra ferramenta é necessária, o ProM na versão 6.14. Este programa receberá o arquivo MXML gerado anteriormente. A Figura 23 ilustra a tela inicial do programa. Para iniciar o processo de análise é necessário a importação do arquivo de mineração, portanto ao clicar no botão *"Import..."* localizado no canto superior direito da tela, uma janela *pop-up* será aberta para realizar a navegação entre diretórios e selecionar o arquivo desejado.



Figura 23 – *Screenshot* da interface da ferramenta ProM.

Após selecionar o arquivo, outra janela *pop-up* será aberta, como ilustra a Figura 24, solicitando que seja selecionado o *plugin* de importação a ser utilizado. Neste trabalho, a opção escolhida foi *"ProM log files (Naive)"*, após isso o arquivo é importado com sucesso.

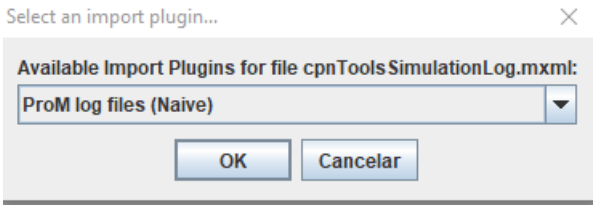


Figura 24 – *Screenshot* da janela de seleção do *plugin* de importação.

Uma vez que o arquivo foi importado, o passo seguinte é selecionar a opção *"Use resource"*, destacada em vermelho na Figura 25. Ao clicar neste botão avança-se para a próxima etapa da mineração de processos, pois agora deve ser escolhido o *plugin* a ser utilizado para realizar a extração das informações.

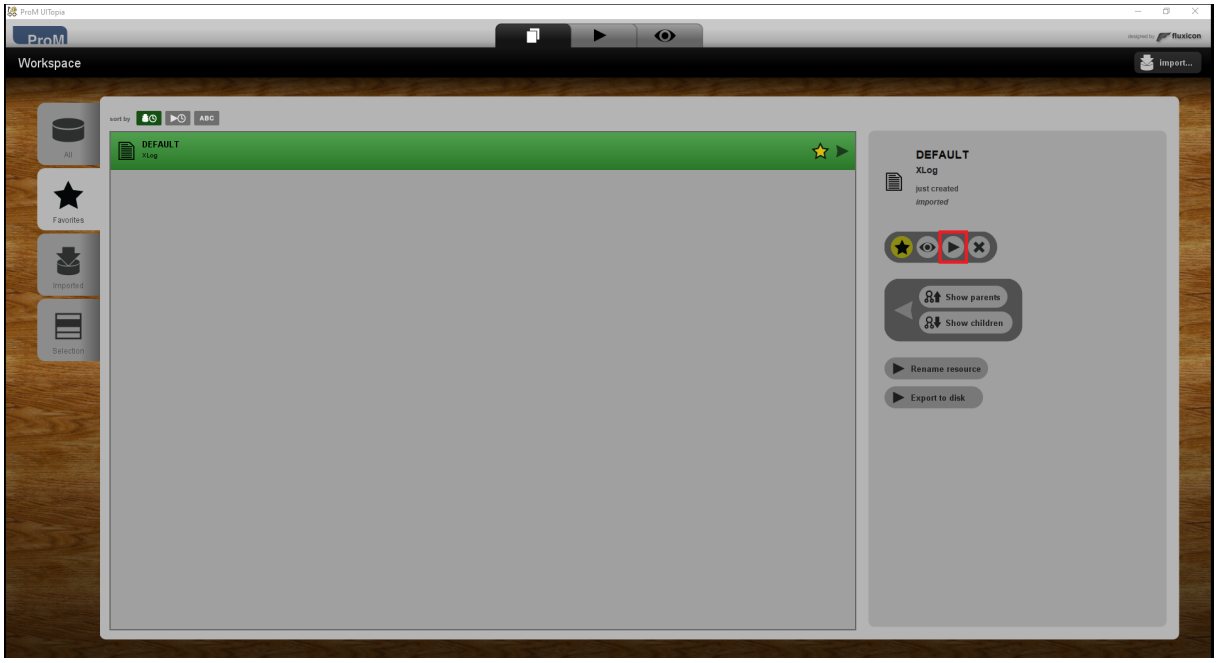


Figura 25 – *Screenshot* da ferramenta ProM após inserir o arquivo no formato MXML.

Em seguida, o programa apresentará a aba de ações, demonstrada através da Figura 26, nela pode ser observado 3 abas principais, sendo elas: *"Input"*, *"Actions"* e *"Output"*. Na aba denominada *Input*, está localizado o arquivo de entrada, fornecendo os dados de entrada em que o *plugin* escolhido deverá atuar.

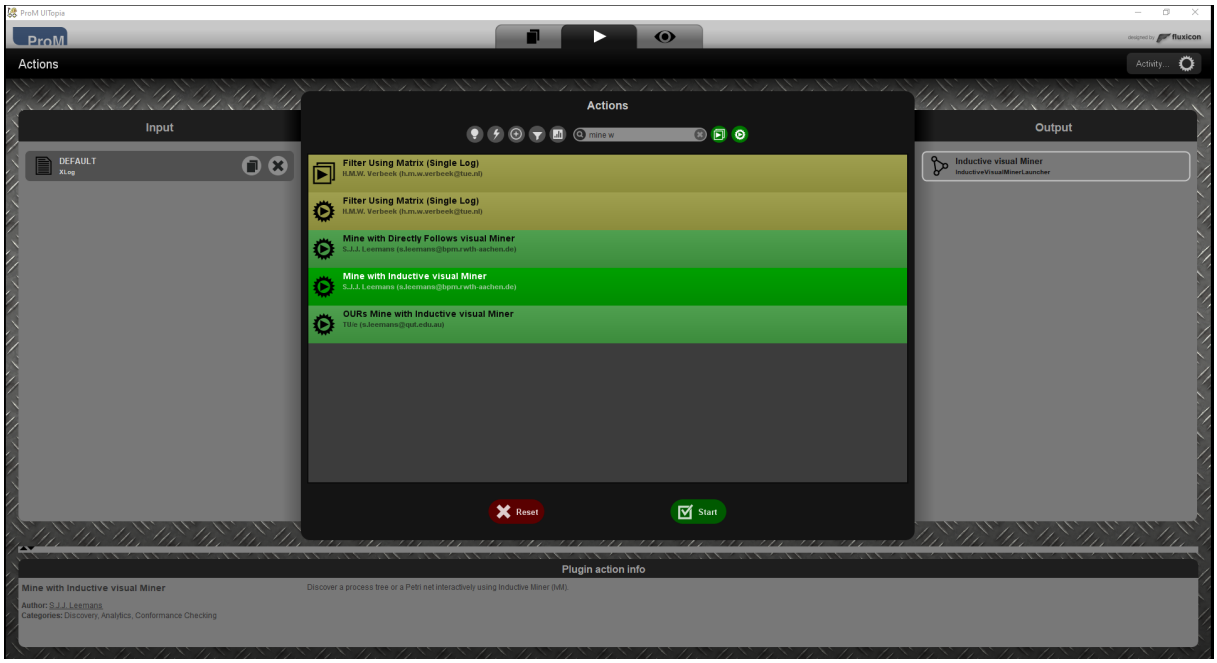


Figura 26 – *Screenshot* da ferramenta ProM durante a escolha do *plugin*.

Já na aba nomeada *"Actions"*, está presente todos os *plugins* instalados automaticamente com a ferramenta ProM, porém vale ressaltar que outro programa denominado

ProM PM 6.14, que também faz parte da instalação da ferramenta, possibilita realizar o gerenciamento dos *plugins* instalados. Para este trabalho, o *plugin* utilizado foi o *"Mine with Inductive Visual Miner"*, conforme demonstrado na Figura 26. Por fim, a aba *"Output"* demonstra a saída gerada pelo *plugin* escolhido. Na parte inferior da tela, na Figura 26 pode ser visualizada algumas informações sobre os *plugins*. A partir disso, o botão *"Start"* deve ser acionado para a execução do *plugin*.

A Figura 27 ilustra o modelo gerado a partir dos *logs* minerados pelo *plugin*. Na parte inferior da tela encontra-se uma barra de reprodução dos casos, que representa a animação da interação dos casos com o modelo, a reprodução pode ser pausada, e a velocidade da animação ser ajustada.

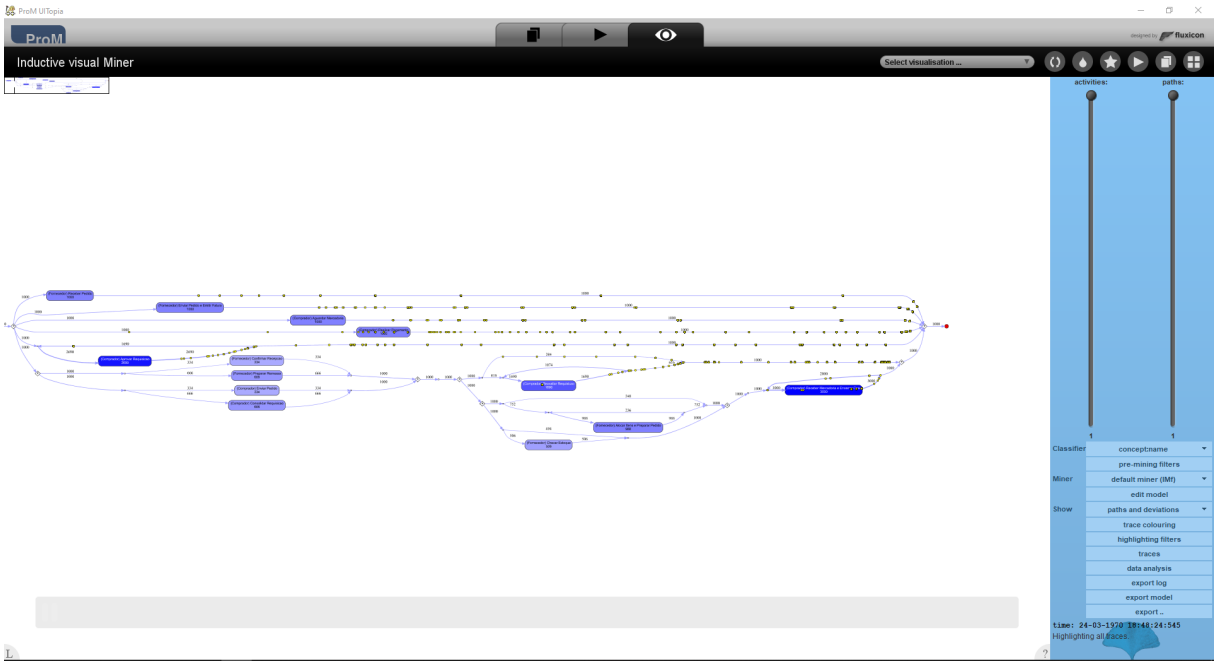


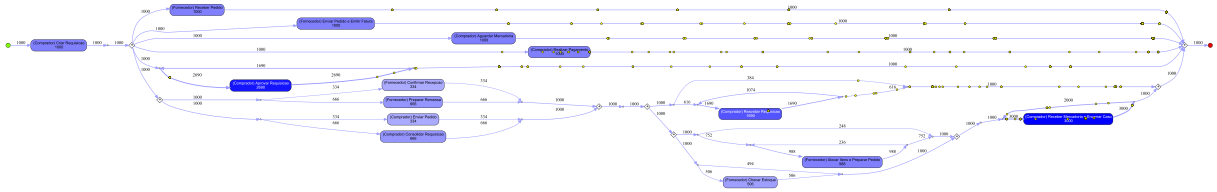
Figura 27 – *Screenshot* da ferramenta ProM ao executar o *plugin* para o modelo da *WorkFlow net* Interorganizacional.

Na ferramenta, é possível identificar duas barras deslizantes, demonstradas pela Figura 28h, onde a barra *"Activities"*, mostra que 100% das atividades estão sendo utilizadas no modelo gerado, isso pode ser concluído através do número *1* abaixo da barra, já que ela vai de 0 a 1 representando 0% a 100%. Já a barra *"Paths"*, demonstra que 100% dos caminhos encontrados através da mineração dos *logs* de eventos, estão sendo exibidos. Ambas as barras podem ser ajustadas e o modelo ilustrado na ferramenta é reajustado em tempo real. Na Figura 28a, é possível destacar diversos elementos presentes no modelo:

- Início do processo: representado pelo círculo verde (Figura 28b).
- Fim do processo: representado pelo círculo vermelho (Figura 28c).

- Escolha exclusiva: representado pelo símbolo de “ou exclusivo”, ou seja, apenas um dos caminhos é tomado (Figura 28d).
- Concorrência: representado pelo símbolo da Figura 28e, é o caso em que se deve executar todas as atividades registradas simultaneamente.
- Intercalação: representado pelo símbolo da Figura 28f, representa a divisão de caminhos, mas permite que apenas um por vez esteja ativo.
- Escolha inclusiva: representa o “ou tradicional” onde divide o caminho em um ou mais ramos subsequentes, representado pela Figura 28g.
- Atividades do processo: representadas pelos “balões” ligados pelos arcos.
- Arcos: mostram quais atividades podem seguir diretamente para cada atividade subsequente.
- Casos: representados pelos círculos amarelos percorrendo o modelo através da animação.
- Números nas atividades e nos arcos: representam a frequência que o comportamento representado ocorreu no evento.

A Figura 28i demonstra as opções de filtros disponíveis para classificação, mineração e visualização no modelo. Neste trabalho foi utilizado alguns elementos de filtros disponíveis no *plugin “Mine with Inductive Visual Miner”*, por se tratar de um modelo com dados simulados. O primeiro elemento a ser destacado se chama seletor de classificador, nele o analista pode escolher de qual perspectiva o modelo gerado pelo *plugin* será construído. No caso do modelo utilizado nesta pesquisa, as opções disponíveis são: categoria, *status*, *name*, *resource* e *timestamp*. No modelo em questão o seletor utilizado foi o *name*, onde cada retângulo arredondado carrega o nome da atividade executada no modelo. Com esse filtro é possivel verificar o fluxo do modelo detectado através dos *logs* de eventos a partir de diferentes perspectivas.



(a) Modelo com 100% dos caminhos visíveis.



(b) Início.



(c) Fim.



(d) Escolha exclusiva.



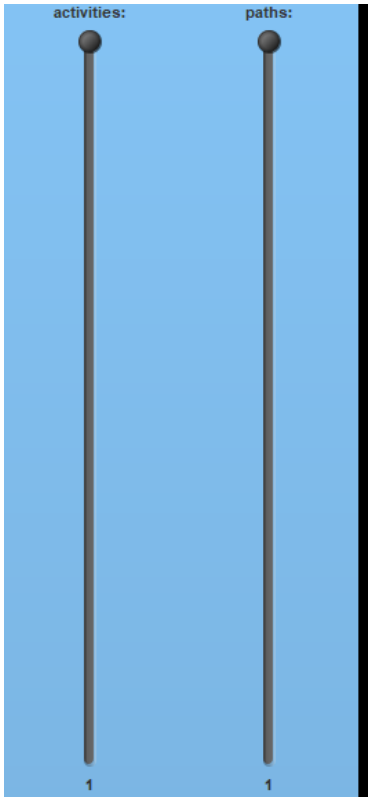
(e) Concorrência.



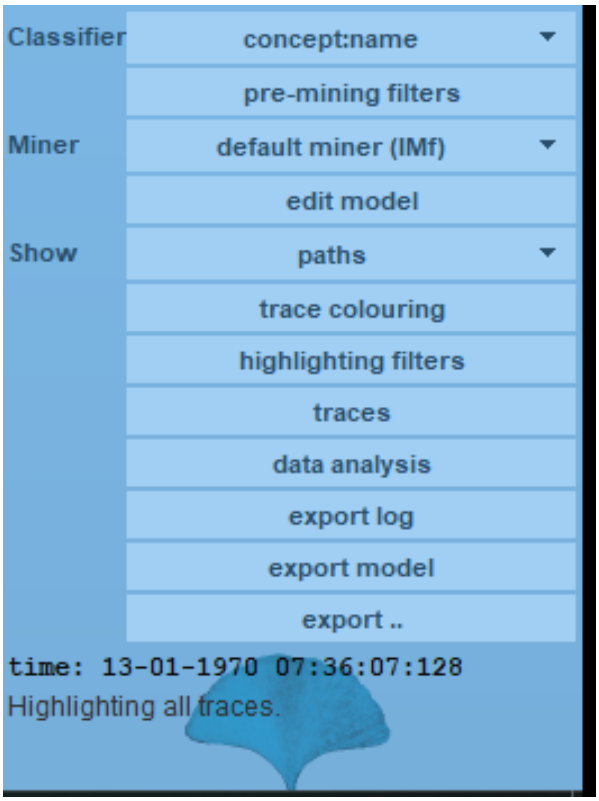
(f) Intercalação.



(g) Escolha inclusiva.



(h) Barras de controle para atividades e caminhos.



(i) Menu lateral com diferentes filtros para aplicação.

Figura 28 – Principais elementos do *launcher* do *plugin*.

Outro elemento interessante para ser utilizado na mineração de processos interorganizacionais, é o seletor de modo de visualização, que permite a adição de vários tipos de informação ao modelo inicial sendo eles:

- **Caminhos**: é o modo padrão do *plugin*, fornece informações básicas sobre o modelo

como o número de ocorrências de cada atividade, e a frequência de cada arco.

- **Caminhos e desvios**: inclui as informações padrão do filtro anterior, porém com a adição das informações sobre os desvios do modelo, contudo essa informação é uma aproximação da realidade. A Figura 29 demonstra os desvios encontrados no modelo com apenas 80% dos caminhos sendo exibidos. Os desvios são representados pelas setas tracejadas em vermelho espalhadas no modelo.

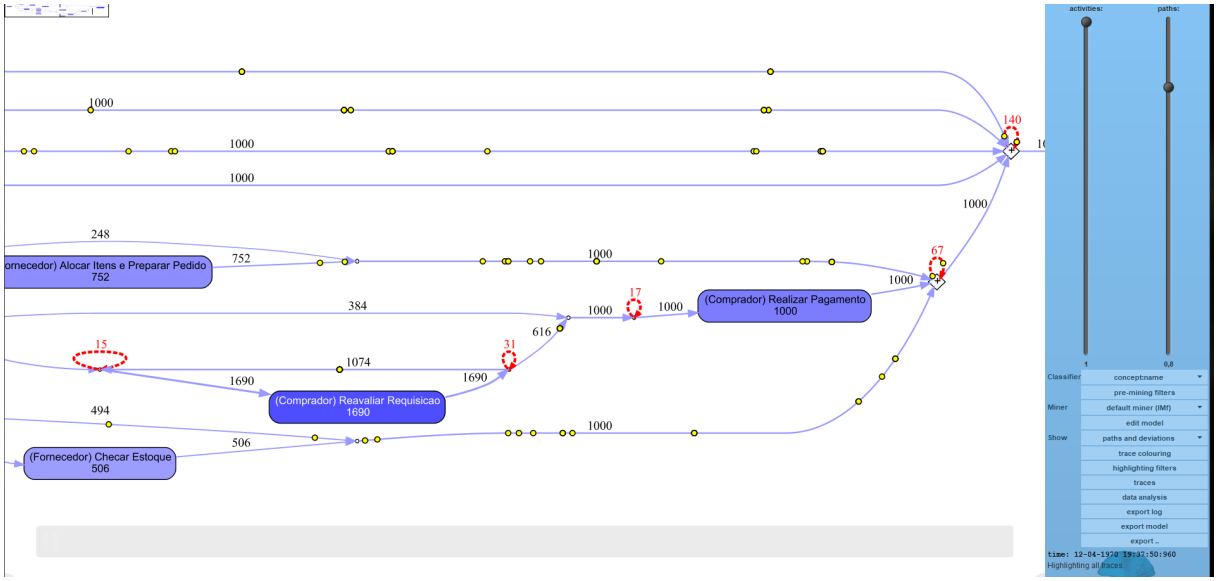


Figura 29 – *Screenshot* do filtro caminhos e desvios.

- **Caminhos e tempos de permanência**: responsável por mostrar o modelo e as atividades, com seu respectivo tempo médio de permanência em cada atividade de forma distinta. Dessa forma é possível identificar as atividades e os processos que demandam mais tempo de execução, onde podem haver gargalos e enfileiramento de casos. Uma maneira de complementar a análise desse filtro parte da animação dos casos sendo executados pelo modelo, pois durante a animação é possível notar o acúmulo e a perda de velocidade dos casos ao passar pelas atividades que demandam mais tempo. A Figura 30 demonstra a atividade com maior tempo médio de execução do modelo analisado. Pode-se observar que a cor das atividades, é vermelha, e intensifica-se de acordo com o tempo que ela demora em média para ser executada, da mesma forma que adquire uma coloração menos intensa quanto menor for esse tempo.

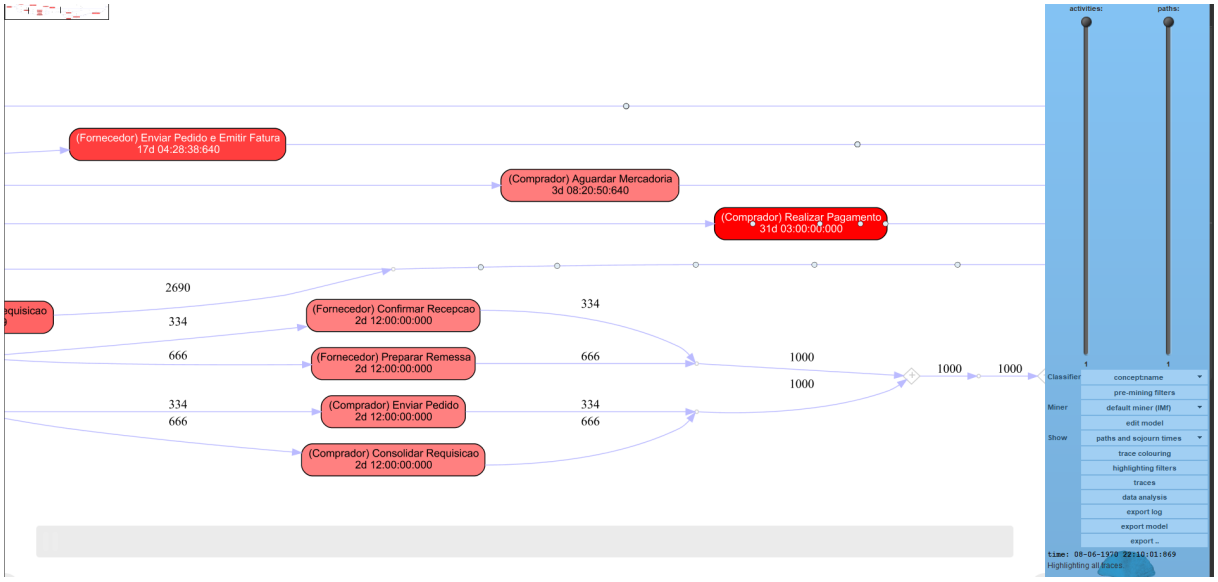


Figura 30 – *Screenshot* do filtro caminhos e tempos de permanência.

Por fim, o alternador da visualização de traço que está localizado na ferramenta na barra lateral direita apenas como *"traces"*, ao ser selecionado abre uma nova janela, conforme ilustra a Figura 31, onde cada caso do arquivo de mineração é representado por uma sequência de blocos coloridos que representam as atividades do processo. Nesse filtro é possível verificar caso a caso, ID por ID a conformidade do modelo, com o registro real dos *logs* realizando a verificação do processo ideal com o real.

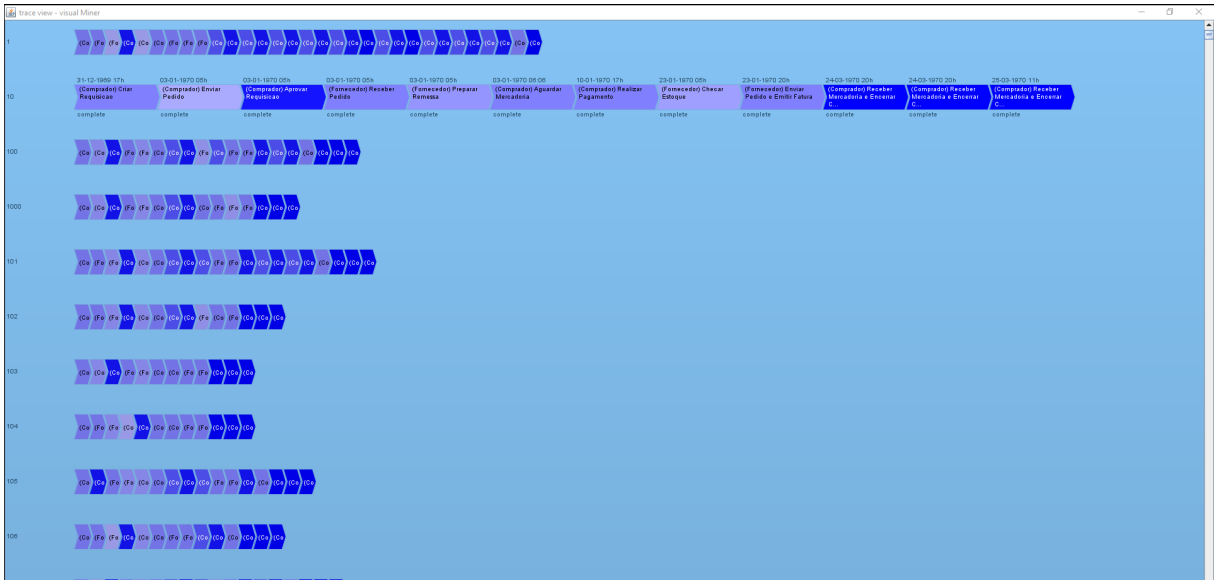


Figura 31 – *Screenshot* do filtro alternador da visualização de traço.

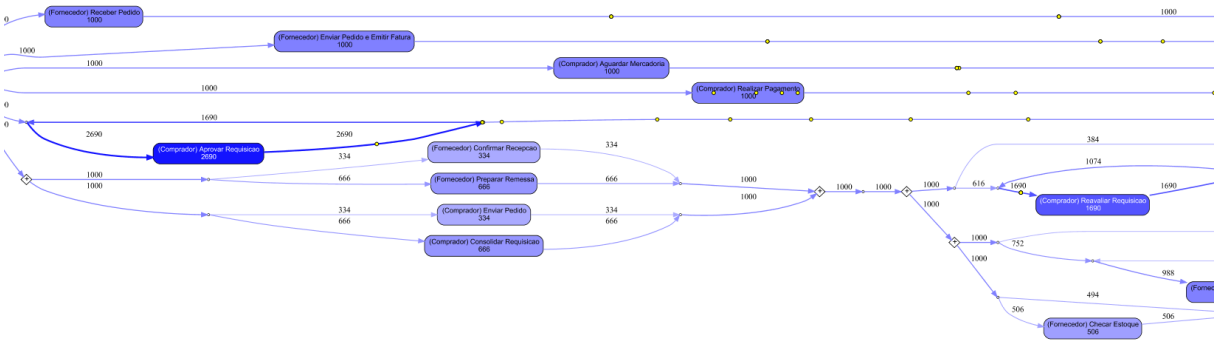
Ao analisar o registro da Figura 31 destaca-se as informações presentes como o número do caso localizado a esquerda da tela, seguido pelos blocos, que carregam o nome de cada atividade executada para cada ID específico. Na parte superior do bloco encontra-se o registro *timestamp* no formato dia, mês, ano e hora. Já na parte inferior de

cada bloco encontra-se o *event type* da atividade. Com essa análise é possível destacar comportamentos incorretos do processo, por exemplo, a realização do pagamento do pedido pelo comprador antes mesmo da fatura ser emitida pelo fornecedor, o que seria um erro em um ambiente de produção real, assim como a preparação da remessa pelo fornecedor antes mesmo da checagem do estoque. Esse filtro faz o mapeamento de cada caso, permitindo a visualização do comportamento do processo em diferentes cenários.

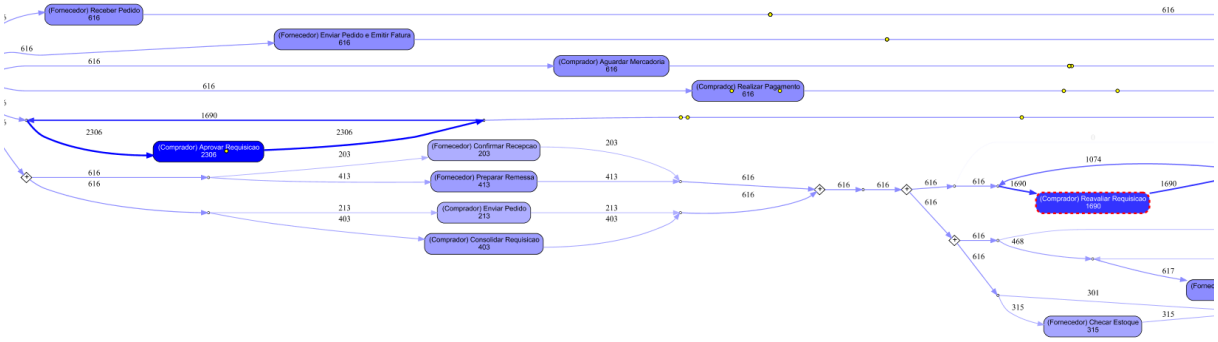
5.1.3 Análise e Resultados

A partir da análise com mineração de processos, foi possível identificar inconsistências no processo, que apenas a ferramenta de simulação de rede de Petri não seria capaz de identificar. Grande parte dos problemas identificados são relativos a má distribuição das atividades no processo, por exemplo, atividades que não fazem um tipo de verificação necessária, ou ainda, atividades que se repetem indefinidamente sem que haja de fato a necessidade de se repetir. No modelo analisado, alguns desses problemas foram:

- falta de verificação na atividade "reavaliação da requisição": nesse problema identifica-se a necessidade de garantir que seja verificado se a requisição também não foi aprovada, pois ao analisar os valores registrados nas frequências das atividades de "aprovar requisição" e "reavaliar requisição", fica claro que a quantidade de registros aprovados se repete para o mesmo caso, pois a atividade "aprovar requisição" foi registrada 2690 vezes, já a atividade reavaliar requisição foi registrada 1690. Quando se analisa a conformidade dessa informação, ao clicar sobre a atividade conforme ilustra a Figura 32b, percebe-se que dos 1690 casos que passaram por reavaliação, apenas 384 não foram realmente aprovados. O que leva a uma inconsistência, já que a atividade tem ao total 2690 ocorrências, como representado na Figura 32a. Ou seja, a maioria dos casos foram reavaliados sem necessidade, causando atrasos no processo, além de desperdício de recursos.



(a) Sem foco na atividade.



(b) Com foco na atividade.

Figura 32 – Análise das atividades de aprovar e reavaliar requisição.

- tempo das atividades: através dos tempos de execução nos *logs* é possível realizar uma análise relacionada ao desempenho dos processos em relação a quantidade de tempo necessária para desempenhar as atividades das organizações. No modelo analisado, apenas para fins de testes, foi implementada uma função de geração de tempo. Porém, com a análise foi possível perceber que isso interfere diretamente no funcionamento do modelo, pois uma atividade com um tempo maior de execução demora mais para avançar e as próximas atividades não podem ser realizadas. Utilizando o filtro “Traces”, demonstrado na Figura 31, é possível verificar as informações de cada caso de maneira isolada do início ao fim do modelo. Com isso foi possível identificar que atividades que deveriam ser de execução obrigatória, não estavam sendo executadas em todos os casos, apenas em alguns. A Figura 33 exemplifica o caso de ID 10, onde pode ser observado que faltam algumas atividades que são obrigatórias na comunicação entre comprador e fornecedor, por exemplo: Confirmar Recepção, Consolidar Requisição, Alocar Itens e Preparar Pedido. Em outros casos, ao se observar o tempo de execução, algumas atividades foram realizadas antes de outras que só deveriam acontecer depois. Estruturalmente, o fluxo do processo está correto, mas analisando os tempos de execução é possível identificar que na prática o processo vai se comportar diferente do esperado.



Figura 33 – *Screenshot* do filtro alternador da visualização de traço de um caso isolado.

Outro ponto a ser destacado após as análises realizadas está relacionado com a forma que o modelo é representado na ferramenta de mineração de processos. No início da pesquisa imaginava-se que os processos interorganizacionais seriam tratados como duas redes distintas pela ferramenta de mineração de processos, fato que se mostrou totalmente diferente, pois o modelo gerado pelo *plugin Inductive Visual Miner* é apenas uma rede envolvendo todos os processos e atividades juntamente com a quantidade de suas ocorrências. Isso reforça que, apesar de ser dois processos locais em organizações diferentes, de fato eles funcionarão como um único processo global, uma vez que as atividades são dependentes.

6 Conclusões e Trabalhos Futuros

6.1 Conclusões

O objetivo dessa pesquisa foi propor um mecanismo de análise baseado na aplicação combinada de redes de Petri e mineração de processos para processos interorganizacionais. A proposta consistiu em criar um modelo de *WorkFlow* interorganizacional na ferramenta CPN Tools e adaptá-lo para gerar *event logs*. A partir disso, os *logs* extraídos com as informações do processo foram submetidos à ferramenta ProM. Com isso, foi possível obter os fluxos e extrair características específicas dos processos.

Com os diversos filtros da ferramenta de mineração foi possível simular o processo e identificar problemas na execução das atividades. O modelo em rede de Petri indicou que estruturalmente o processo estava correto. No entanto, a mineração indicou que os problemas estavam na execução das atividades, deixando o processo com inconsistências significativas.

Por exemplo, a análise de conformidade revelou que as atividades não estavam seguindo o fluxo que deveriam. Foi possível observar que as atividades não estavam sendo executadas simultaneamente, como o esperado, e sim de forma excludente. Também foi possível analisar o tempo de execução do processo. Por se tratar de um modelo interorganizacional, é natural que o início da execução de uma atividade esteja vinculado ao término da execução de outra. Entretanto, na análise foi possível observar, de acordo com os tempos de execução, que algumas atividades executavam fora do tempo. Por exemplo, a atividade de realizar o pagamento era executada antes mesmo do pedido ser faturado.

Dessa forma, a junção da rede de Petri com a mineração de processo se mostrou uma alternativa interessante para analisar processos interorganizacionais. Com a rede de Petri, é possível visualizar os processos locais e a interação entre eles, formando assim o processo global. Já a mineração aponta as inconsistências na execução das atividades por meio do rastreamento dos casos simulados. Enquanto a mineração aponta as inconsistências semânticas, a rede de Petri permite que os processos sejam tratados separadamente. Com isso, é possível corrigir pontualmente as falhas que geram prejuízos ao processo como um todo.

6.2 Trabalhos futuros

Esta pesquisa foi realizada com dados fictícios e simulados, portanto não reflete a imprevisibilidade de processos e dados reais de mercado, das diversas implicações do dia a

dia. Logo, seria interessante considerar a aplicação do método proposto em um processo interorganizacional real. Durante o desenvolvimento deste trabalho, foi possível identificar que há diferentes ferramentas de mineração de processos sendo utilizadas nas organizações. Seria interessante então explorar outras ferramentas disponíveis para a análise de processos interorganizacionais..

Referências

AALST, W. *Verification of Workflow nets*. 1997. 407-426 p. Citado na página 25.

AALST, W. Interorganizational workflows: An approach based on message sequence charts and petri nets. *Systems Analysis, Modelling Simulation*, v. 34, 01 1999. Citado 6 vezes nas páginas 11, 15, 26, 27, 45 e 46.

AALST, W. M. P. van der. *Structural characterizations of sound workflow nets*. Eindhoven, 1996. Citado na página 25.

AALST, W. M. P. van der. The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, v. 8, p. 21–66, 02 1998. Citado 2 vezes nas páginas 25 e 41.

AALST, W. M. P. van der. Intra- and inter-organizational process mining: Discovering processes within and between organizations. In: JOHANNESSON, P.; KROGSTIE, J.; OPDAHL, A. L. (Ed.). *The Practice of Enterprise Modeling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 1–11. ISBN 978-3-642-24849-8. Citado 2 vezes nas páginas 42 e 43.

AALST, W. M. P. van der. *Process mining : discovery, conformance and enhancement of business processes*. Germany: Springer, 2011. ISBN 978-3-642-19344-6. Citado na página 43.

AALST, W. M. P. van der et al. Workflow mining: A survey of issues and approaches. *Data & Knowledge Engineering*, v. 47, n. 2, p. 237–267, nov 2003. Citado 3 vezes nas páginas 15, 36 e 37.

AALST, W. M. P. van der; VERBEEK, H. M. W.; HAUSCHILDT, D. A petri-net-based tool to analyze workflows. In: FARWER, B.; MOLDT, D.; STEHR, M. (Ed.). *Proceedings of Petri Nets in System Engineering (PNSE'97)*. Hamburg: University of Hamburg, 1997. p. 78–90. Citado na página 21.

AALST, W. M. P. van der; WEIJTERS, A. J. M. M.; MARUSTER, L. Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, v. 16, n. 9, p. 1128–1142, oct 2004. Citado na página 36.

AALST, W. van der. *Process Mining: Data Science in Action*. 2. ed. Berlin, Heidelberg: Springer-Verlag, 2016. ISBN 978-3-662-49850-7. Citado 5 vezes nas páginas 15, 36, 37, 38 e 39.

AALST, W. van der; HEE, K. van. *Workflow Management: Models, methods and systems*. Cambridge, MA: MIT Press, 2004. Citado 5 vezes nas páginas 15, 21, 24, 25 e 26.

AALST, W. van der; STAHL, C. *Modeling Business Processes: A Petri Net-Oriented Approach*. Cambridge, Massachusetts; London, England: The MIT Press, 2011. Citado 2 vezes nas páginas 21 e 25.

AOUACHRIA, M. et al. A process mining method for inter-organizational business process integration. *ACM Trans. Manage. Inf. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 15, n. 1, mar. 2024. ISSN 2158-656X. Disponível em: <https://doi.org/10.1145/3638062>. Citado 3 vezes nas páginas 15, 43 e 44.

CARDOSO, J.; VALETTE, R. *Redes de Petri*. [S.l.]: Editora da UFSC, 1997. Citado 3 vezes nas páginas 17, 18 e 19.

DataBridge Market Research. *Global Process Mining Market*. 2024. <https://www.databridgemarketresearch.com/reports/global-process-mining-market>. Citado na página 14.

ENGEL, R. et al. Analyzing inter-organizational business processes. *Inf. Syst. E-Bus. Manag.*, Springer-Verlag, Berlin, Heidelberg, v. 14, n. 3, p. 577–612, ago. 2016. ISSN 1617-9846. Disponível em: <https://doi.org/10.1007/s10257-015-0295-2>. Citado 3 vezes nas páginas 15, 42 e 44.

IEEE Task Force on Process Mining. *XES - eXtensible Event Stream*. 2025. <https://www.xes-standard.org/>. Acesso em: 4 jun. 2025. Citado na página 40.

JENSEN, K. An introduction to the practical use of coloured petri nets. In: *Advanced Course on Petri Nets*. [S.l.: s.n.], 1996. p. 237–292. ISBN 978-3-540-65307-3. Citado na página 31.

JENSEN, K.; KRISTENSEN, L. *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. [S.l.]: Springer, 2009. ISBN 978-3-642-00283-0. Citado 4 vezes nas páginas 31, 32, 33 e 36.

MarketsandMarkets. *Process Mining Market*. 2023. <https://www.marketsandmarkets.com/Market-Reports/process-mining-market-176608355.html>. Citado na página 14.

MEDEIROS, A.; GüNTHER, C. *Process mining: Using CPN tools to create test logs for mining algorithms*. 2005. Acesso em: 26 mar. 2025. Disponível em: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=131bddfb9edb4bf20c9f54429db2a5a3c63e3917>. Citado 4 vezes nas páginas 15, 41, 43 e 57.

MURATA, T. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, v. 77, n. 4, p. 541–580, abr. 1989. Citado 3 vezes nas páginas 15, 17 e 19.

NOCE, F. S. *Uma abordagem para a modelagem, análise e simulação de processos de negócios interorganizacionais baseada nas workflow nets e na ferramenta CPN Tools*. 2017. Acesso em: 26 mar. 2025. Disponível em: <https://repositorio.ufu.br/handle/123456789/19353>. Citado 2 vezes nas páginas 41 e 43.

OECD. *Global value and supply chains*. 2024. <https://www.oecd.org/en/topics/policy-issues/global-value-and-supply-chains.html>. Citado na página 14.

Outsource Accelerator. *Outsourcing Statistics*. 2023. <https://www.outsourceaccelerator.com/articles/outsourcing-statistics/>. Citado na página 14.

Process Excellence Network. *Highlights of Gartner's 2025 Magic Quadrant for Process Mining Platforms*. 2024. <https://www.processexcellencenetwork.com/process-mining/articles/highlights-gartners-2025-magic-quadrant-process-mining>. Citado na página 14.

Relações Exteriores. *A Nova Divisão Internacional do Trabalho*. 2025. <https://relacoesexteriores.com.br/nova-divisao-trabalho-internacional/>. Citado na página 14.

ROZINAT, A.; AALST, W. M. P. van der. Conformance checking of processes based on monitoring real behavior. *Information Systems*, v. 33, n. 1, p. 64–95, mar 2008. Citado na página 38.